Deep Multi-task Learning for Railway Track Inspection

Xavier Gibert, *Member, IEEE,* Vishal M. Patel, *Senior Member, IEEE,* and Rama Chellappa, *Fellow, IEEE*

***Abstract*—Railroad tracks need to be periodically inspected and monitored to ensure safe transportation. Automated track inspection using computer vision and pattern recognition methods have recently shown the potential to improve safety by allowing for more frequent inspections while reducing human errors. Achieving full automation is still very challenging due to the number of different possible failure modes as well as the broad range of image variations that can potentially trigger false alarms. Also, the number of defective components is very small, so not many training examples are available for the machine to learn a robust anomaly detector. In this paper, we show that detection performance can be improved by combining multiple detectors within a multi-task learning framework. We show that this approach results in improved accuracy for detecting defects on railway ties and fasteners.**

***Index Terms*—Railway track inspection, Multi-task Learning, Deep Convolutional Neural Networks, Material Identification.**

I. Introduction

MONITORING the condition of railway components is essential to ensure train safety, especially on High Speed Rail (HSR) corridors. Amtrak's recent experience with concrete ties has shown that they have different kind of problems than wood ties [1]. The locations and names of the basic track elements mentioned in this paper are shown in Figure 1. Although concrete ties have life expectancies of up to 50 years, they may fail prematurely for a variety of reasons, such as the result of alkali-silicone reaction (ASR) [2] or delayed ettringite formation [3]. ASR is a chemical reaction between cement alkalis and non-crystalline (amorphous) silica. This forms alkali-silica gel at the aggregate surface. These reaction rims have a very strong affinity with water and have a tendency to swell. These compounds can produce internal pressures that are strong enough to create cracks, allowing moisture to penetrate, and thus accelerating the rate of deterioration. Delayed Ettringite Formation (DEF) is a type of internal sulfate attack that occurs in concrete that has been cured at excessively high temperatures. In addition to ASR and DEF, ties can also develop fatigue cracks due to normal traffic or by being impacted by flying debris or track maintenance machinery. Once small cracks develop, repeated cycles of freezing and thawing will eventually lead to bigger defects.

Xavier Gibert is with X (formerly known as Google[x]), Mountain View, CA 94043 USA (e-mail: xgibert@google.com). This research was performed while the first author was a Ph.D. student at the University of Maryland.

Rama Chellappa is with the Department of Electrical Engineering and the Center for Automation Research, UMIACS, University of Maryland, College Park, MD 20742 USA (e-mail: rama@umiacs.umd.edu)

Vishal M. Patel is with the Department of Electrical Engineering, Rutgers University, Piscataway, NJ 08854 USA (e-mail: vishal.m.patel@rutgers.edu)

Manuscript received .....

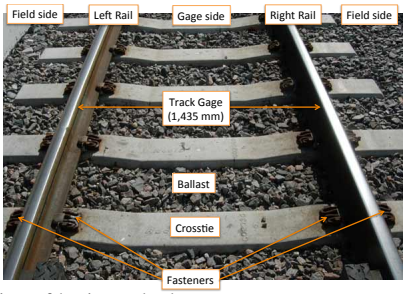


Fig. 1. Definition of basic track elements.

Fasteners maintain gage by keeping both rails firmly attached to the crossties. According to the Federal Railroad Administration (FRA) safety database[1], in 2013, out of 651 derailments due to track problems, 27 of them were attributed to gage widening caused by defective spikes or rail fasteners, and another 2 to defective or missing spikes or rail fasteners. Also, in the United States, regulations enforced by the FRA[2] prescribe visual inspection of high-speed rail tracks with a frequency of once or twice per week, depending on track speed. These manual inspections are currently performed by railroad personnel, either by walking on the tracks or by riding a hi-rail vehicle at very low speeds. However, such inspections are subjective and do not produce an auditable visual record. In addition, railroads usually perform automated track inspections with specialized track geometry measurement vehicles at intervals of 30 days or less between inspections. These automated inspections can directly detect gage widening conditions. However, it is preferable to find fastening problems before they develop into gage widening conditions.

Recent advances in CMOS imaging technology, have resulted in commercial-grade line-scan cameras that are capable of capturing images at resolutions of up to 4,096×2 and line rates of up to 140 KHz. At the same time, high-intensity LED-based illuminators with life expectancies in the range of 50,000 hours are commercially available. This technology enables virtually maintenance-free operation over several months. Therefore, technology that enables autonomous visual track inspection from an unattended vehicle (such as a passenger train) may become a reality in the not-too-distant future. In our previous works [4] and [5], we showed that it is possible to automatically inspect the condition of ties and fasteners. In this paper, we extend these techniques and integrate them in a multi-task learning framework. This combined system achieves better performance than learning each task separately.

[1] http://safetydata.fra.dot.gov

[2] 49 CFR 213 – Track Safety Standards

This paper is organized as follows. Related works on inspection of railway tracks using computer vision are discussed in section II. The problem addressed in this paper is described in section III. The overall approach and system architecture is presented in section IV. Material classification, segmentation and tie assessment algorithm is described in section V. Fastener detection and assessment algorithm is described in section VI. Experimental results are presented in section VII, and section VIII concludes the paper with a brief summary and discussion.

II. Related Works

Railway Track Inspection: Since the pioneering work by Trosino *et al.* [6], [7], machine vision technology has been gradually adopted by the railway industry as a track inspection technology. Those first generation systems were capable of collecting images of the railway right of way and storing them for later review, but they did not facilitate any automated detection. As faster processing hardware became available, several vendors began to introduce vision systems with increasing automation capabilities.

In [8], [9], Marino *et al.* describe their VISyR system, which detects hexagonal-headed bolts using two 3-layer neural networks (NN) running in parallel. Both networks take the 2-level discrete wavelet transform (DWT) of a 24×100 pixel sliding window (their images use non-square pixels) as an input to generate a binary output indicating the presence of a fastener. The difference is that the first NN uses Daubechies wavelets, while the second one uses Haar wavelets. This wavelet decomposition is equivalent to performing edge detection at different scales with two different filters. Both neural networks are trained with same examples. The final decision is made using the maximum output of each neural network.

In [10], [11], Gibert *et al.* describe their VisiRail system for joint bar inspection. The system is capable of collecting images on each rail side, and finding cracks on joint bars using edge detection and a Support Vector Machine (SVM) classifier that analyzes features extracted from these edges. In [12], Babenko describes a fastener detection method based on a convolutional filter bank that is applied directly to intensity images. Each type of fastener has a single filter associated with it, whose coefficients are calculated using an illumination-normalized version of the Optimal Tradeoff Maximum Average Correlation Height (OT-MACH) filter [13]. This approach allowed accurate fastener detection and localization and achieved over 90% fastener detection rate on a dataset of 2,436 images. However, the detector was not tested on longer sections of track. In [14], Resendiz *et al.* use texture classification using a bank of Gabor filters followed by an SVM to determine the location of rail components such as crossties and turnouts. They also use the MUSIC algorithm to find spectral signatures to determine expected component locations. In [15], Li *et al.* describe a system for detecting tie plates and spikes. Their method, which is described in more detail in [16], uses an AdaBoost-based object detector [17] with a model selection mechanism that assigns the object class which produces the highest number of detections within a window of 50 frames. Table I summarizes several systems reported in the literature.

Convolutional Neural Networks: The idea of enforcing translation invariance in neural networks using weight sharing goes back to Fukushima's Neocognitron [27]. Based on this idea, LeCun *et al.* developed the architecture of Deep Convolutional Neural Networks (DCNN) and used it for digit recognition [28], and later for more general optical character recognition (OCR) [29]. During the last few years, DCNNs have become ubiquitous in achieving state-of-the-art results in image classification [30], [31] and object detection [32]. This resurgence of DCNNs has been facilitated by the availability of efficient GPU implementations and open source libraries such as Caffe [33] and Torch7 [34]. More recently, DCNNs have been used for semantic image segmentation. For example, the work of [35] shows how a DCNN can be converted to a Fully Convolutional Network (FCN) by replacing fully-connected layers with convolutional ones. In the context of intelligent transportation systems, DCNN-based methods have been applied to the problem of traffic flow prediction in [36] and [37].

Multi-task Learning: Multi-task learning (MTL) is an inductive transfer learning technique in which two or more learning machines are trained cooperatively [38]. It is a generalization of multi-label learning in which each training sample has only been labeled for one of the tasks. In MTL settings there is a mechanism in which knowledge learned for one task is transferred to the other tasks [39]. The idea is that each task can benefit by reusing knowledge that has been learned while training for the other tasks. Backpropagation has been recognized as an effective method for learning distributed representations [40]. For instance, in multitask neural networks, we jointly minimize one global loss function

$$\Phi = \sum_{t=1}^{T} \lambda_t \sum_{i=1}^{N_t} E_t \left(f(x_{ti}), y_{ti} \right), \tag{1}$$

where T is the number of tasks, N_t is the number of training samples for task t, y_{ti} is the ground truth label for training sample x_{ti}, f is the the multi-output function computed by the network, and E_t is the loss function for task t. This contrasts with the Single Task Learning (STL) setting, in which we minimize T independent loss functions

$$\Phi_t = \sum_{i=1}^{N_t} E_t \left(f_t(x_{ti}), y_{ti} \right), \quad t \in \{1 \dots T\}. \tag{2}$$

In MTL, the weighting factor λ_t is necessary to compensate for imbalances in the complexity of the different tasks and the amount of training data available. When using backpropagation, it is necessary to adjust λ_t's to ensure that all tasks are learning at optimal rates.

One-shot Learning: To achieve good generalization performance, traditional machine learning methods require a minimum number of training examples from each class. This is necessary for the machine to learn a model that can handle variations in image appearance that result from changes in illumination, scale, rotation, background clutter, and so on. However, the occurrence of each type of anomaly is very

TABLE I
EVOLUTION OF AUTOMATED VISUAL RAILWAY COMPONENT INSPECTION METHODS.

Authors	Year	Components	Defects	Features	Decision methods
Stella *et al.* [9], [18], [19]	2002–09	Fasteners	Missing	DWT	3-layer NN
Singh *et al.* [20]	2006	Fasteners	Missing	Edge density	Threshold
Hsieh *et al.* [21]	2007	Fasteners	Broken	DWT	Threshold
Gibert *et al.* [10], [11]	2007–08	Joint Bars	Cracks	Edges	SVM
Babenko [12]	2008	Fasteners	Missing/Defective	Intensity	OT-MACH corr.
Xia *et al.* [22]	2010	Fasteners	Broken	Haar	Adaboost
Yang *et al.* [23]	2011	Fasteners	Missing	Direction Field	Correlation
Resendiz *et al.* [14]	2013	Ties/Turnouts	–	Gabor	SVM
Li *et al.* [15]	2014	Tie plates	Missing spikes	Lines/Haar	Adaboost
Feng *et al.* [24]	2014	Fasteners	Missing/Defective	Haar	PGM
Gibert *et al.* [25]	2014	Concrete ties	Cracks	DST	Iterative shrinkage
Khan *et al.* [26]	2014	Fasteners	Defective	Harris-Stephen, Shi-Tomasi	Matching errors
Gibert *et al.* [4]	2015	Fasteners	Missing/Defective	HOG	SVM
Gibert *et al.* [5]	2015	Concrete ties	Tie Condition	Intensity	Deep CNN

infrequent, so in anomaly detection settings it is only possible to find one or a few number of examples from which to learn from. If we try to learn a complete model for a new class using such a limited number of examples, this model would overfit and would not be able to generalize to new data. However, if we reuse knowledge that has been learned while learning other related classes, we can learn better models. This is known as one-shot learnig [41]. We pose this one-shot learning problem as a special case of multi-task learning, in which one task consists of learning the abundant classes, while the other task learns the uncommon classes. Both tasks share a common low-level representation because all fasteners are built with common materials. In this paper, we train an auxiliary network on a 5-class fastener classification using more than 300K fasteners for the sole purpose of learning a good representation that regularizes the broken fastener detector.

III. PROBLEM DESCRIPTION

The application described in this paper consists of inspecting railroad tracks for defects on crossties and rail fasteners using single-view line-scan cameras. The crossties may of different materials (e.g. wood, concrete, plastic, or metal), and the fasteners could be of different types (e.g. elastic clips, bolts, or spikes). We have posed this problem as two detection problems: object detection (good, broken, or missing fastener), and semantic segmentation (chips and crumbling concrete ties and other material classes).

A. Dataset

The dataset used to demonstrate this approach comprises 85 miles of track in which the bounding boxes of 203,287 ties have been provided. This data is very challenging to work with. The images were collected from a moving vehicle and although there was artificial illumination, there are significant variations in illumination due to sun position and shadows. To reduce friction between rails and wheels and prolong their usable lives, railroads may lubricate them using special equipment mounted along the tracks. At locations near these lubricators, tracks get dirty and the accumulation of greasy deposits significantly change the appearance of the images. There are also some spots in which the tracks are covered by mud being pumped through the ballast during heavy rainfall. Moreover, there are also places in which the ballast is unevenly distributed and pieces of ballast rock cover the ties and fasteners being inspected. Also, leaves, weeds, branches, trash and other debris may occlude the track components being inspected.

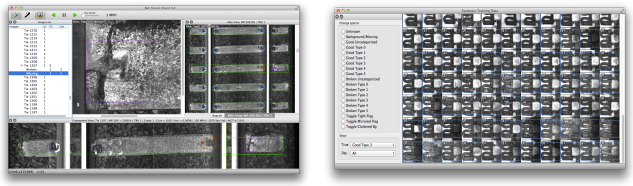


Fig. 2. GUI tool used to generate the training set and to review the detection results.

B. Data Annotation

Due to the large size of this dataset, we implemented a customized software tool that allows the user to efficiently visualize and annotate the data (see Figure 2 for a screenshot). This tool has been implemented in C++ using the Qt framework and communicates with the data repository through the secure HTTPS protocol, so it can be used from any computer with an Internet connection without having to set up tunnel or VPN connections. The tool allows assigning a material category to each tie as well as its bounding box. The tool also allows defining polygons enclosing regions containing crumbling, chips or ballast. The tool also allows the user to change the threshold of the defect detector and select a subset of the data for display and review. It also has the capability of exporting lists of detected defects as well as summaries of fastener inventories by mile.

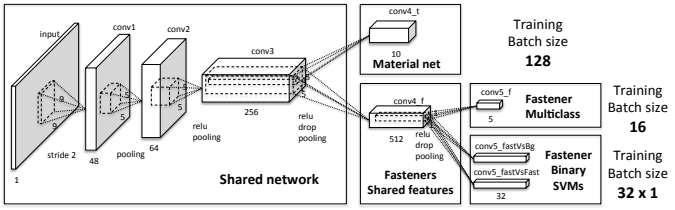


Fig. 3. Network architecture.

IV. Approach

A. Overall Architecture

Our design is a Fully Convolutional Network [35] based on the architecture that we introduced in [5]. That network was trained with 10 classes of materials and produces feature maps with 10 different channels. In this paper, we extend that architecture by adding two additional branches to the network. The first one is a coarse-level fastener classifier trained on a large number of examples. The second branch produces 32 binary outputs. These outputs correspond to the same binary SVMs that we used in our previous version of the detector [4] described in more detail in section VI.

The implementation is based on the BVLC Caffe framework [33]. For the material classification task, we have a total of 4 convolutional layers between the input and the output layer, while for fastener detection tasks we have 5 convolutional layers. The first three layers are shared among all the tasks. The fasteners task is, in turn, divided in two subtasks: coarse-level and fine-grained classification (see section VI for more details). The network uses rectified linear units (ReLU) as non-linear activation functions, and overlapping max pooling units of size 3×3. All max pooling units have a stride of 2, except the one on top of that has a stride of 1. We use dropout [42] regularization on layer 3 (with a ratio of 0.1) and layer 4 on the fasteners branch (with a ratio of 0.2). The network also uses weight decay regularization. On the fasteners branch, we increase the weight decay factors on layers 4 and 5 by $10\times$ and $100\times$ respectively to reduce overfitting.

We first apply global gain normalization on the raw image to reduce the intensity variation across the image. This gain is calculated by smoothing the signal envelope estimated using a median filter. We estimate the signal envelope by low-pass filtering the image with a Gaussian kernel. As DCNNs are robust to illumination changes, normalizing the image to make the signal dynamic range more uniform improves accuracy and convergence speed. We also subtract the mean intensity value, which is calculated on the whole training set. The network architecture is illustrated in Figure 3.

B. Training Procedure

To generate our training set, we initially selected $\sim$30 good quality (with no occlusion and clean edges) samples from each object category at random from the whole repository and annotated the bounding box location and object class for each of them. Our training software also automatically picks, using a randomly generated offset, a background patch adjacent to each of the selected samples. Once we had enough samples from each class, we trained binary classifiers for each of the classes against the background and tested on the whole dataset. Then, we randomly selected misclassified samples and added those that had good or acceptable quality (no occlusion) to the training set. To maintain the balance of the training set, we also added, for each difficult sample, 2 or 3 neighboring samples. Since there are special types of fasteners that do not occur very frequently (such as the c-clips or j-clips used around joint bars), in order to keep the number of samples of each type in the training set as balanced as possible, we added as many of these infrequent types as we could find.

Careful annotation of the dataset resulted in the training set of 2819 fully-annotated fasteners. Moreover, some of the classes had very few examples. For instance, there are only 28 broken fast-clips, and just 38 j-clips in the dataset. If we just had used this limited data, we would not have been able to learn a good representation. Fortunately, both of these two uncommon classes of fasteners share parts with the other ones. Therefore, if we can make layer *conv4_f* learn a good model for fastener parts, layer *conv5_f* would be able to learn how to distinguish between fasteners by combining such parts, even if the number of training examples is limited.

Therefore, we created an auxiliary fastener data set. Since the only purpose of this dataset is to help learn parts, we just used the bounding boxes and labels automatically generated by our previous detector [4], whose error rate is just 0.37%. We sampled 62,500 fasteners from each of 5 coarse classes. The first class contains missing and broken fasteners, the next 3 classes contain fasteners corresponding to each of the classes containing the most samples (PR-clips, e-clips, and fast-clips), and the last class contains everything else.

We train the network using stochastic gradient descent on mini-batches of 128 image patches of size 75×75 plus 48 fastener images of 182×182. The fastener images include 16 from the auxiliary fastener dataset and 1 from each of the binary SVM tasks. We do data augmentation on material classification by randomly mirroring vertically and/or horizontally the training samples. The patches are cropped randomly among all regions that contain the texture of interest. To increase robustness against adverse environment conditions, such as rain, grease or mud, we identified images containing such difficult cases and automatically resampled the data so that at least 50% of the data is sampled from such difficult images. We do data augmentation on fasteners by randomly mirroring vertically the symmetric classes and randomly cropping the fastener offset uniformly distributed within a +/-9 pixel range in both directions.

V. Material Identification and Segmentation

A. Architecture

The material classification task at layer *conv4_t* generates ten score maps at 1/16th. Each value $\Phi_i(x, y)$ in the score map corresponds to the likelihood that pixel location (x, y) contains material of class i. The ten classes of materials are defined in Figure 8.

B. Score Calculation

To detect whether an image contains a broken tie, we first calculate the scores at each site as

$$S_b(x,y) = \max_{i \notin \mathcal{B}} \Phi_i(x,y) - \Phi_b(x,y) \tag{3}$$

where $b \in \mathcal{B}$ is a defect class (crumbling or chip). Then we calculate the score for the whole image as

$$S_b = \frac{1}{\beta - \alpha} \int_{\alpha}^{\beta} \widehat{F}^{-1}(t) dt \tag{4}$$

where $\widehat{F}^{-1}$ refers to the t sample quantile calculated from all scores $S_b(x,y)$ in the image. The detector reports an alarm if $S > \tau$, where τ is the detection threshold. We used $\alpha = 0.9$ and $\beta = 1$.

VI. Fasteners Assessment

In this section, we describe the details of the fastener assessment task. Figure 4 shows the types of defects that our algorithm can detect.

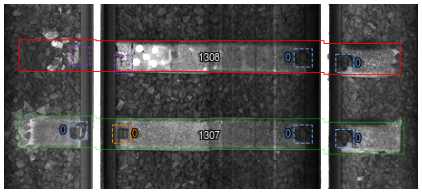


Fig. 4. Example of defects that our algorithm can detect. Blue boxes indicate good fasteners, orange boxes indicate broken fasteners, and purple boxes indicate missing fasteners. White numbers indicate tie index from last mile post. Other numbers indicate type of fastener (for example, 0 is for e-clip fastener).

A. Overview

Due to surface variations that result from grease, rust and other elements in the outdoor environment, segmentation of railway components is a very difficult task. Therefore, we avoid it by using a detector based on a sliding window that we run over the inspectable area of the tie. The features learned at layer *conv4_f* are computed from the shared features at *conv3*. The reason for sharing the features with the material classification task is that there is overlap between both tasks. For instance, the material classification task needs to learn to distinguish between fasteners and the other materials, regardless of the type of fastener. Also, the fastener detection class needs to discriminate between fasteners and background, regardless of the type of background. In our previous work, we used the Histogram of Oriented Gradients (HOG) [43] descriptor for detecting fasteners. Although the results that we obtained using HOG features were better than previously proposed methods, this approach still has its limitations. For instance, the dimensionality of the feature vector is very large (12,996), consuming a lot of memory and computational resources, and the linear classifier cannot handle occlusions very well. Therefore, in this paper we attempt to learn the features by training the network end to end.

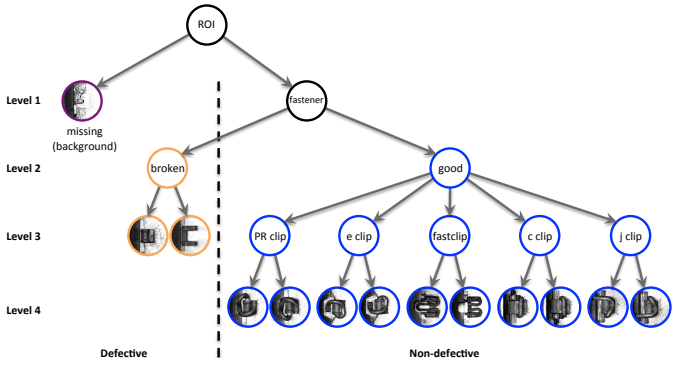


Fig. 5. Object categories used for detection and classification (from coarsest to finest levels).

B. Classification

Our goal is to simultaneously detect, within each predefined Region of Interest (ROI), the most likely fastener location and then classify such detections into one of three basic conditions: background (or missing fastener), broken fastener, and good fastener. Then, for good and broken fastener conditions, we want to assign class labels for each fastener type (PR clip, e-clip, fastclip, c-clip, and j-clip). Figure 5 shows the complete categorization that we use, from coarsest to finest. At the coarsest level, we want to classify fastener vs. unstructured background clutter. The background class also includes images of ties where fasteners are completely missing. We have done this for these reasons: 1) it is very difficult to train a detector to find the small hole left on the tie after the whole fastener has been ripped off, 2) we do not have enough training examples of missing fasteners, and 3) most missing fasteners are on crumbled ties for which the hole is no longer visible. Once we detect the most likely fastener location, we want to classify the detected fastener between broken vs. good, and then classify it into the most likely fastener type. Although this top-down reasoning works for a human inspector, it does not work accurately in a computer vision system because both the background class and the fastener class have too much intra-class variations. As a result, we have resorted to a bottom-up approach.

To achieve the best possible generalization at test time, we have based our detector on the maximum margin principle of the SVM. The SVM separating hyperplane is obtained by minimizing the regularized hinge loss function,

$$E = \sum_i \max(0, 1 - y_i(w \cdot x_i + b)) + \frac{\lambda}{2}\|w\|, \tag{5}$$

where $x_i \in \mathbb{R}^{512}$ are the outputs of layer *conv4_f* and $y_i \in \{-1, +1\}$ their corresponding ground truth labels (whose meaning will be explain later). The gradients with respect to the parameters w and b are

$$\frac{\partial E}{\partial w} = -\sum_i y_i x_i \delta[y_i(w \cdot x_i + b) < 1] + \lambda w \tag{6}$$

$$\frac{\partial E}{\partial b} = -\sum_i y_i \delta[y_i(w \cdot x_i + b) < 1], \tag{7}$$

where $\delta\{\text{condition}\}$ is 1 if condition is true and -1 otherwise. The gradient of the hinge loss function with respect to the data (which is back-propagated down to the lower layers) is

$$\frac{\partial E}{\partial x_i} = -y_i w \delta[y_i(w \cdot x_i + b) < 1]. \quad (8)$$

Once the parameters converge, these gradients become highly sparse and only the difficult training samples contribute to to updating the parameters on layer *conv4_f* and all the layers below.

Instead of training a multi-class SVM, we use the one-vs-rest strategy, and instead of treating the background class as just another object class, we treat it as a special case and use a pair of SVMs per object class. For instance, if we had used a single learning machine, we would be forcing the classifier to perform two different unrelated tasks: a) reject that the image patch does not contain random texture and b) reject that the object does not belong to the given category. Therefore, given a set of object classes $\mathcal{C}$, we train two detectors for each object category. The first one, with weights b_c, classifies each object class $c \in \mathcal{C}$ vs. the background/missing class $m \notin \mathcal{C}$, and the second one, with weights f_c classifies object class c vs. other object classes $\mathcal{C} \backslash c$. As illustrated in Figure 6, asking our linear classifier to perform both tasks at the same time would result in a narrower margin than training separate classifiers for each individual task. Moreover, to avoid rejecting cases where all f_c classifiers produce negative responses, but one or more b_c classifiers produce strong positive responses that would otherwise indicate the presence of a fastener, we use the negative output of f_c as a soft penalty. Then the likelihood that sample x belongs to class c becomes

$$L_c(x) = b_c \cdot x + \min(0, f_c \cdot x), \quad (9)$$

where $x = HOG(I)$ is the feature vector extracted from a given image patch I. The likelihood that our search region contains at least one object of class c is the score of the union, so

$$L_c = \max_{x \in \mathcal{X}} L_c(x), \quad (10)$$

where $\mathcal{X}$ is the set of all feature vectors extracted within the search region, and our classification rule becomes

$$\hat{c} = \begin{cases} \arg\max_{c \in \mathcal{C}} L_c & \max_{c \in \mathcal{C}} L_c > 0 \\ m & \text{otherwise.} \end{cases} \quad (11)$$

C. Score Calculation

For the practical applicability of our detector, it needs to output a scalar value that can be compared to a user-selectable threshold τ. Since there are several ways for a fastener to be defective (either missing or broken), we need to generate a single score that informs the user how confident the system is that the image contains a fastener in good condition. This score is generated by combining the output of the binary classifiers introduced in the previous section.

We denote the subset of classes corresponding to good fasteners as $\mathcal{G}$ and that of broken fasteners as $\mathcal{B}$. These two subsets are mutually exclusive, so $\mathcal{C} = \mathcal{G} \cup \mathcal{B}$ and $\mathcal{G} \cap \mathcal{B} = \emptyset$.

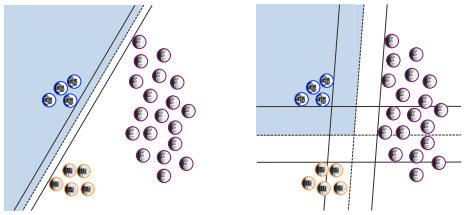


Fig. 6. Justification for using two classifiers for each object category. Shaded decision region corresponds fastener in good condition, while white region corresponds to defective fastener. Blue circles are good fasteners, orange circles are broken fasteners, and purple circles are background/missing fasteners. Left: Classification region of good fastener vs. rest Right: Classification region of intersection of good fastener vs. background and good fastener vs. rest-minus-background. The margin is much wider than possible when a single classifier is used.

To build the score function, we first compute the score for rejecting the missing fastener hypothesis (i.e, the likelihood that there is at least one sample $x \in \mathcal{X}$ such that $x \notin m$) as

$$S_m = \max_{c \in \mathcal{G}} L_c, \quad (12)$$

where L_c is the likelihood of class c defined in (10). Similarly, we compute the score for rejecting the broken fastener hypothesis (i.e, the likelihood that for each sample $x \in \mathcal{X}$, $x \notin \mathcal{B}$) as

$$S_b = -\max_{c \in \mathcal{B}} \max_{x \in \mathcal{X}} f_c \cdot x, \quad (13)$$

The reason why the S_b does not depend on a c-vs-background classifier b_c is because mistakes between missing and broken fastener classes do not need to be penalized. Therefore, S_b need only produce low scores when x matches at least one of the models in $\mathcal{B}$. The negative sign in S_b results from the convention that a fastener in good condition should have a large positive score. The final score becomes the intersection of these two scores

$$S = \min(S_m, S_b). \quad (14)$$

The final decision is done by reporting the fastener as good if $S > \tau$, and defective otherwise.

D. Training Procedure

The advantage of using a maximum-margin classifier is that once we have enough support vectors for a particular class, it is not necessary to add more inliers to improve classification performance. Therefore, we can potentially achieve relatively good performance with only having to annotate a very small fraction of the data.

E. Alignment Procedure

For learning the most effective object detection models, the importance of properly aligning the training samples cannot be emphasized enough. Misalignment between different training samples would cause unnecessary intra-class variation that would degrade detection performance. Therefore, all the training bounding boxes were manually annotated, as tightly as

Fig. 7. CTIV platform used to collect the images.

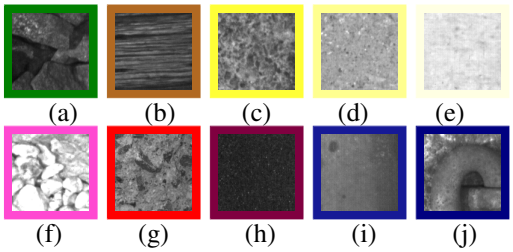


Fig. 8. Material categories. (a) ballast (b) wood (c) rough concrete (d) medium concrete (e) smooth concrete (f) crumbling concrete (g) chipped concrete (h) lubricator (i) rail (j) fastener

possible to the object contour by the same person to avoid inducing any annotation bias. For training the fastener vs. background detectors, our software cropped the training samples using a detection window centered around these boxes. For training the fastener vs. rest detectors, our software centered the positive samples using the human-generated annotation, but the negative samples were re-centered to the position where the fastener vs. background detector generated the highest response. This was done to force the learning machine to learn to differentiate object categories by taking into account parts that are not common across object categories.

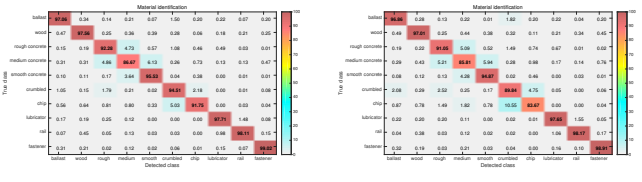
VII. Experimental Results

To evaluate the accuracy of our fastener detector, we have tested it on 85 miles of continuous trackbed images. These images were collected on the US Northeast Corridor (NEC) by ENSCO Rail's Comprehensive Track Inspection Vehicle (CTIV) (see Figure 7). The CTIV is a hi-rail vehicle (a road vehicle that can also travel on railway tracks) equipped with several track inspection technologies, including a Track Component Imaging System (TCIS). The TCIS collects images of the trackbed using 4 Basler sprint (spL2048-70km) linescan cameras and a custom line scan lighting solution.

The sprint cameras are based on CMOS technology and use a CameraLink interface to stream the data to a rack-mounted computer. Each camera contains a sensor with 2 rows of 2,048 pixels that can sample at line rates of up to 70KHz. The cameras can be set to run in dual-line mode (high-resolution) or in "binned" mode, where the values of each pair of pixels are averaged inside the sensor. The cameras were set up in binned mode so each camera generated a combined row of 2,048 pixels at a line rate of 1 line/0.43mm. The sampling rate was controlled by the signal generated from a BEI distance encoder mounted on one of the wheels. The camera positions and optics were selected to cover the whole track with minimal perspective distortion and their fields of view had some overlap to facilitate stitching.

The collected images were automatically stitched together and saved into several files, each containing a 1-mile image. These files were preprocessed by ENSCO Rail using their proprietary tie detection software to extract the boundary of all the ties in each image. We verified that the tie boundaries were accurate after visually correcting invalid tie detections. We downsampled the images by a factor of 2, for a pixel size of 0.86 mm. To assess the detection performance under different operating conditions, we flagged special track sections where the fastener visible area was less than 50% due to a variety of occluding conditions, such as protecting covers for track-mounted equipment or ballast accumulated on the top of the tie. We also flagged turnouts so we could report separate ROC curves for both including and excluding them. All the ties in this dataset are made of reinforced concrete, were manufactured by either San-Vel or Rocla, and were installed between 1978 and 2010.

For a fair comparison between the approach proposed in this paper and previously published results, we trained the algorithm with the same dataset and annotations that we used in our previous works described in [5] and [4]. We used the output of our previous fastener detection algorithm [4] to extract new fastener examples for semisupervised learning.



(a) (b)

Fig. 9. Confusion matrix of material classification on 2.5 million 80×80 image patches with Deep Convolutional Neural Networks using (a) multi-task learning (b) single task learning [5].

A. Material Identification

We divided the dataset into 5 splits and used 80% of the images for training and 20% for testing and we generated a model for each of the 5 possible training sets. For each split of the data, we randomly sampled 50,000 patches of each class. Therefore, each model was trained with 2 million patches. We trained the network using a batch size of 128 for a total of 300,000 iterations with a momentum of 0.9 and a weight decay of 5×10^{-5}. The learning rate is initially set to 0.01 and it decays by a factor of 0.5 every 30,000 iterations. The following methods are compared in this paper:

- **Deep CNN MTL 3:** The method described in Section V with the full architecture in Figure 3.
- **Deep CNN MTL 2:** The previous method without the binary SVM subnet.

- **Deep CNN STL:** The previous method without the fasteners subnet and a batch size of 64. This single task learning baseline is exactly the same model used in [5].
- **LBP-HF with approximate Nearest Neighbor:** The Local Binary Pattern Histogram Fourier descriptor introduced in [44] is invariant to global image rotations while preserving local information. We used the implementation provided by the authors. To perform approximate nearest neighbor we used FLANN [45] with the 'autotune' parameter set to a target precision of 70%.
- **Uniform LBP with approximate Nearest Neighbor** The $LBP^{u2}_{8,1}$ descriptor [46] with FLANN.
- **Gabor features with approximate Nearest Neighbor:** We filtered each image with a filter bank of 40 filters (4 scales and 8 orientations) designed using the code from [47]. As proposed in [48], we compute the mean and standard deviation of the output of each filter and build a feature descriptor as $f = [\mu_{00}\ \sigma_{00}\ y_{01}\ \ldots\ \mu_{47}\ \sigma_{47}]$. Then, we perform approximate nearest neighbor using FLANN with the same parameters.

The material classification results are summarized in Table II and the confusion matrices are shown in Figure 9.

TABLE II
MATERIAL CLASSIFICATION RESULTS.

Method	Accuracy
Deep CNN MTL 3	**95.02**%
Deep CNN MTL 2	93.60%
Deep CNN STL [5]	93.35%
LBP-HF with FLANN	82.05%
$LBP^{u2}_{8,1}$ with FLANN	82.70%
Gabor with FLANN	75.63%

Since we are using a fully convolutional DCNN, we directly transfer the parameters learned using small patches to a network that takes one 4096×320 image as an input, and generates 10 score maps of dimension 252×16 each. The segmentation map is generated by taking the label corresponding to the maximum score. Figure 11 shows several examples of concrete and wood ties, with and without defects and their corresponding segmentation maps.

B. Crumbling and Chipped Tie Detection

The first 3 rows in Figure 11 show examples of a crumbling ties and their corresponding segmentation map. Similarly, rows 4 through 6 show examples of chipped ties. To evaluate the accuracy of the crumbling and chipped tie detector described in Section V-B we divide each tie in 4 images and evaluate the score (4) on each image independently. Due to the large variation in the area affected by crumbling/chip we assigned a severity level to each ground truth defect, and for each severity level we plot the ROC curve of finding a defect when ignoring lower level defects. The severity levels are defined as the ratio of the inspectable area that is labeled as a defect. Figure 10 shows the ROC curves for each type of anomaly. Because of the choice of the fixed $\alpha = 0.9$ in (4) the performance is not reliable for defects under 10% severity. For defects that are bigger than the 10% threshold, at a false positive rate (FPR) of 10 FP/mile the true positive rates (TPR) are 89.42% for crumbling and 93.42% for chips. This is an improvement of 3.36% and 1.31% compared to the STL results reported in [5]. The results on chipped tie detection are mixed, while there is an improvement at 2 FP/mile, the detection performance at 10 FP/mile is lower than that of STL. Table III summarizes the results.

TABLE III
TIE CONDITION DETECTION. FOR CHIPPED AND CRUMBLING, ONLY TIES WITH AT LEAST 10% AFFECTED AREA ARE INCLUDED. FASTENER RATES CORRESPOND INCLUDE THOSE FOR WHICH THE TRACK IS CLEAR.

Condition	FPR	MTL	STL
Crumbling Tie	10 FP/mile	**89.42**%	86.54%
	2 FP/mile	**82.21**%	74.52%
Chipped Tie	10 FP/mile	92.76%	**94.08**%
	2 FP/mile	**90.13**%	88.52%
Fastener	10 FP/mile	**99.91**%	98.41%
	2 FP/mile	**96.74**%	93.19%

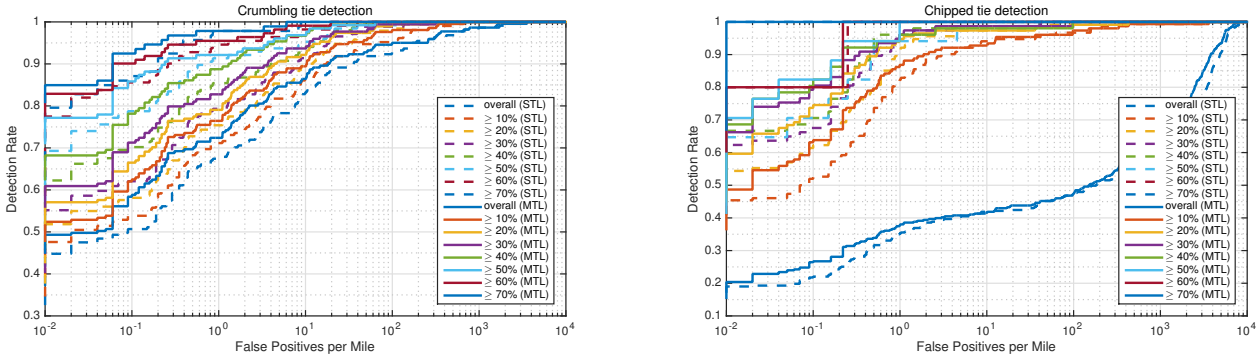
C. Fastener Categorization

On our dataset, we have a total of 8 object categories (2 for broken clips, 1 for PR clips, 1 for e-clips, 2 for fast clips, 1 for c-clips, and 1 for j-clips) plus a special category for the background (which includes missing fasteners). We also have 4 synthetically generated categories by mirroring non-symmetric object classes (PR, e, c, and j clips), so we use a total of 12 object categories at test time.

For training our detectors, we used the same training set as in [4], which has a total of 3,805 image patches, including 1,069 good fasteners, 714 broken fasteners, 33 missing fasteners, and 1,989 patches of background texture. During training, we also included the mirrored versions of the missing/background patches and all symmetric object classes.

In addition to the proposed method described in Section VI, we have also implemented and evaluated the following alternative methods:

- **STL (WACV 2015):** The method in [4] uses the same scores as the proposed method, based on HOG features instead of the features learned at layer *conv4_f*.
- **Intensity normalized OT-MACH:** As in [12], for each image patch, we subtract the mean and normalize the image vector to unit norm. For each class c, we design an OT-MACH filter in the Fourier domain using $h_c = [\alpha I + (1-\alpha)D_c]^{-1}\overline{x}_c$ with $\alpha = 0.95$, where I is the identity matrix, $D_c = (1/N_c)\sum_{i=1}^{N_c} x_{ci}x^*_{ci}$, and N_c is the number of training samples of class c.
- **HOG features with OT-MACH:** The method in [12], but replacing intensity with HOG feature. Since HOG features are already intensity-invariant, the design of the filters reduces to $h_c = \overline{x}_c$.
- **HOG features with DAG-SVM:** We run one-vs-one SVM classifiers in sequence. We first run each class against the background on each candidate region. If at least one classifier indicates that the patch is not



(a) (b)

Fig. 10. (a) ROC curve for detecting crumbling tie conditions. (a) ROC curve for detecting chip tie conditions. Each curve is generated considering conditions at or above a certain severity level. Note: False positive rates are estimated assuming an average of 10^4 images per mile. Confusion between chipped and crumbling defects are not counted as false positives.

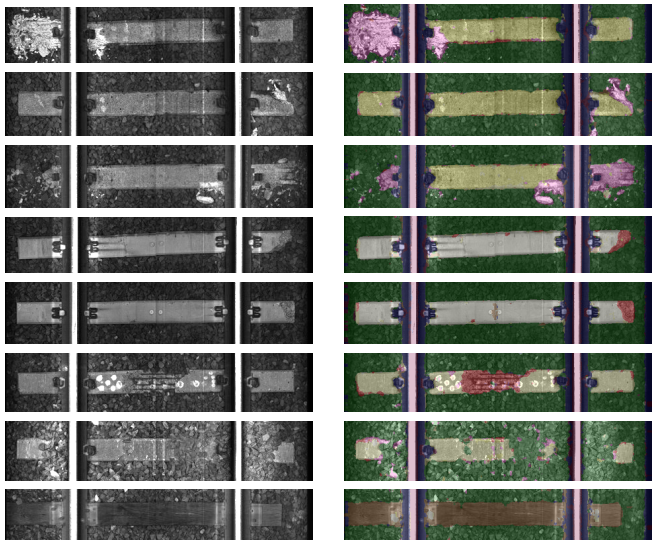


Fig. 11. Semantic segmentation results (images displayed at 1/16 of original resolution). See Figure 8 for color legend.

background, then we run the DAG-SVM algorithm [49] over the remaining classes.

- **HOG features with majority voting SVM:** We run all possible one-vs-one SVM classifiers and select the class with the maximum number of votes.

For the second and third methods, we calculate the score using the formulation introduced in sections VI-B and VI-C, but with $b_c = h_c$ and $f_c = h_c - \sum_{i \neq c} h_i/(C-1)$. For the forth and last methods, we first estimate the most likely class in $\mathcal{G}$ and in $\mathcal{B}$. Then, we set S_b as the output of the classifier between these two classes, and S_m as the output of the classifier between the background and the most likely class.

We can observe in Figure 12 (a) that the proposed method is the most accurate, followed by WACV 2015 STL baseline and HOG with OT-MACH method. The other methods perform poorly on this dataset. In the third row of Table III we compare the fastener detection performance of MTL with the STL baseline.

D. Defect Detection

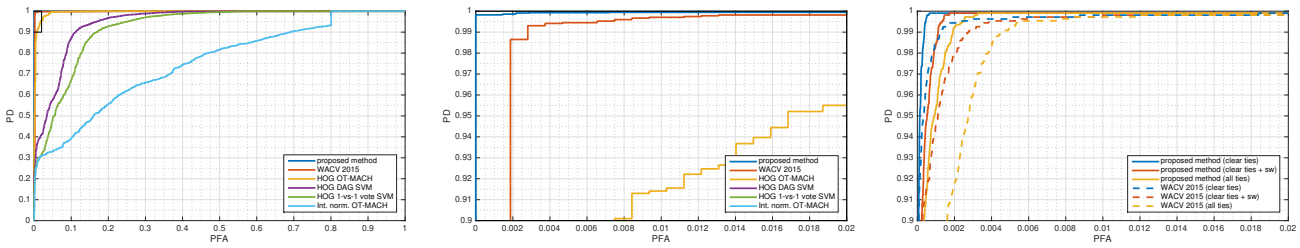
To evaluate the performance of our defect detector, we divided each tie into 4 regions of interest (left field, left gage, right gage, right field) and calculated the score defined by (14) for each of them. Figure 12 shows the ROC curve for crossvalidation on the training set as well as for the testing set of 813,148 ROIs (203,287 ties). The test set contains 1,052 ties images with at least one defective fastener per tie. The total number of defective fasteners in the testing set was 1,087 (0.13% of all the fasteners), including 22 completely missing fasteners and 1,065 broken fasteners. The number of ties that we flagged as "uninspectable" is 2,524 (1,093 on switches, 350 on lubricators, 795 covered in ballast, and 286 with other issues).

We used the ROC on clear ties (blue curve) in Figure 12 (b) to determine the optimal threshold to achieve a design false alarm rate of 0.07% ($\tau = 0.1070$). This target is a bit lower than the 0.1% that we used in the for the baseline experiments. The reason for lowering the sensitivity is that the detection rate pateaus at PFA $> 0.06\%$. Using this sensitivity level, we ran our defective fastener detector at the tie level (by taking the minimum score across all 4 regions). Results are shown in Table IV.

TABLE IV
RESULTS FOR DETECTION OF TIES WITH AT LEAST ONE DEFECTIVE FASTENER.

Subset	Total	# Bad	PD		PFA	
			MTL	STL	MTL	STL
clear ties	200,763	1,037	**99.90%**	98.36%	**0.25%**	0.38%
clear + sw.	201,856	1,045	**99.90%**	97.99%	**0.61%**	0.71%
all ties	203,287	1,052	**99.90%**	98.00%	**1.01%**	1.23%

At this sensitivity level, our MTL detector only misses one defect (compared to 17 type II errors with the baseline detector). The false alarm rate on clear ties goes down to 0.25%, which is 34% lower than the baseline. Figure 13 shows the single defective fastener that was missed. It could be argued that the clip is still holding the rail in place, so it is a very close call.



(a) (a) detail (b)

Fig. 12. ROC curves for the task of detecting defective (missing or broken) fasteners (a) using 5-fold cross-validation on the training set (b) on the 85-mile testing set.

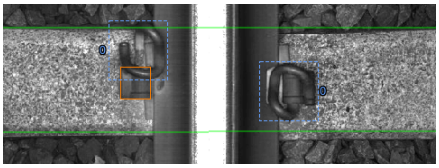


Fig. 13. The single defect missed by our detector. Solid bounding boxes correspond to ground truth annotations. Dashed bounding boxes correspond to the output of the detector. The number 0 corresponds to the PR-clip class, which is correctly classified. The clip has not completely popped out.

VIII. Conclusion and Future Work

This paper has introduced a new algorithm for inspecting railway ties and fasteners that takes advantage of the inherent structure of this problem. We have been able to benefit from scalability advantage of deep convolutional neural networks despite the limited amount of training data in some of the classes. This has been possible by setting up multiple tasks and cooperatively training a shared representation that is effective on each of them. We have showed that not only is possible save computation time by reusing the computation of intermediate features, but also that this representation results in better generalization performance than traditional features.

Acknowledgment

This work was partially supported by the Federal Railroad Administration under contract DTFR53-13-C-00032. The authors thank Amtrak, ENSCO, Inc. and the Federal Railroad Administration for providing the data used in this paper. The authors sincerely thank University of Maryland student Daniel Bogachek for his help setting up earlier crumbling tie detection experiments during his visit at the Center for Automation Research in summer 2014.

References

[1] J. A. Smak, "Evolution of Amtrak's concrete crosstie and fastening system program," in *International Concrete Crosstie and Fastening System Symposium*, June 2012.

[2] M. H. Shehata and M. D. Thomas, "The effect of fly ash composition on the expansion of concrete due to alkalisilica reaction," *Cement and Concrete Research*, vol. 30, pp. 1063–1072, 2000.

[3] S. Sahu and N. Thaulow, "Delayed ettringite formation in swedish concrete railroad ties," *Cement and Concrete Research*, vol. 34, pp. 1675–1681, 2004.

[4] X. Gibert, V. M. Patel, and R. Chellappa, "Robust fastener detection for autonomous visual railway track inspection," in *IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2015.

[5] ——, "Material classification and semantic segmentation of railway track images with deep convolutional neural networks," in *IEEE International Conference on Image Processing (ICIP)*, 2015.

[6] J. Cunningham, A. Shaw, and M. Trosino, "Automated track inspection vehicle and method," May 2000, uS Patent 6,064,428.

[7] M. Trosino, J. Cunningham, and A. Shaw, "Automated track inspection vehicle and method," March 2002, uS Patent 6,356,299.

[8] F. Marino, A. Distante, P. Mazzeo, and E. Stella, "A real-time visual inspection system for railway maintenance: Automatic hexagonal-headed bolts detection," *IEEE Trans. on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, vol. 37, no. 3, pp. 418–428, 2007.

[9] P. De Ruvo, A. Distante, E. Stella, and F. Marino, "A GPU-based vision system for real time detection of fastening elements in railway inspection," in *IEEE International Conference on Image Processing (ICIP)*. IEEE, 2009, pp. 2333–2336.

[10] X. Gibert, A. Berry, C. Diaz, W. Jordan, B. Nejikovsky, and A. Tajaddini, "A machine vision system for automated joint bar inspection from a moving rail vehicle," in *ASME/IEEE Joint Rail Conference & Internal Combustion Engine Spring Technical Conference*, 2007, pp. 289–296.

[11] A. Berry, B. Nejikovsky, X. Gibert, and A. Tajaddini, "High speed video inspection of joint bars using advanced image collection and processing techniques," in *Proc. of World Congress on Railway Research*, 2008.

[12] P. Babenko, "Visual inspection of railroad tracks," Ph.D. dissertation, University of Central Florida, 2009. [Online]. Available: http://crcv.ucf.edu/papers/theses/Babenko_Pavel.pdf

[13] A. Mahalanobis, B. V. K. V. Kumar, S. Song, S. R. F. Sims, and J. F. Epperson, "Unconstrained correlation filters," *Appl. Opt.*, vol. 33, no. 17, pp. 3751–3759, Jun 1994. [Online]. Available: http://ao.osa.org/abstract.cfm?URI=ao-33-17-3751

[14] E. Resendiz, J. Hart, and N. Ahuja, "Automated visual inspection of railroad tracks," *IEEE Trans. on Intelligent Transportation Systems*, vol. 14, no. 2, pp. 751–760, June 2013.

[15] Y. Li, H. Trinh, N. Haas, C. Otto, and S. Pankanti, "Rail component detection, optimization, and assessment for automatic rail track inspection," *IEEE Trans. on Intelligent Transportation Systems*, vol. 15, no. 2, pp. 760–770, April 2014.

[16] H. Trinh, N. Haas, Y. Li, C. Otto, and S. Pankanti, "Enhanced rail component detection and consolidation for rail track inspection," in *IEEE Workshop on Applications of Computer Vision (WACV)*, 2012, pp. 289–295.

[17] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, vol. 1, 2001, pp. I–511–I–518 vol.1.

[18] E. Stella, P. Mazzeo, M. Nitti, C. Cicirelli, A. Distante, and T. D'Orazio, "Visual recognition of missing fastening elements for railroad maintenance," in *IEEE International Conference on Intelligent Transportation Systems*, 2002, pp. 94–99.

[19] F. Marino, A. Distante, P. L. Mazzeo, and E. Stella, "A real-time visual inspection system for railway maintenance: automatic hexagonal-headed bolts detection," *IEEE Trans. on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, vol. 37, no. 3, pp. 418–428, 2007.

[20] M. Singh, S. Singh, J. Jaiswal, and J. Hempshall, "Autonomous rail track inspection using vision based system," in *IEEE International Conference on Computational Intelligence for Homeland Security and Personal Safety*, Oct 2006, pp. 56–59.

[21] H.-Y. Hsieh, N. Chen, and C.-L. Liao, "Visual recognition system of elastic rail clips for mass rapid transit systems," in *ASME/IEEE*

Joint Rail Conference & Internal Combustion Engine Spring Technical Conference, 2007, pp. 319–325.

[22] Y. Xia, F. Xie, and Z. Jiang, "Broken railway fastener detection based on adaboost algorithm," in *IEEE International Conference on Optoelectronics and Image Processing (ICOIP)*, vol. 1. IEEE, 2010, pp. 313–316.

[23] J. Yang, W. Tao, M. Liu, Y. Zhang, H. Zhang, and H. Zhao, "An efficient direction field-based method for the detection of fasteners on high-speed railways," *Sensors*, vol. 11, no. 8, pp. 7364–7381, 2011.

[24] H. Feng, Z. Jiang, F. Xie, P. Yang, J. Shi, and L. Chen, "Automatic fastener classification and defect detection in vision-based railway inspection systems," *IEEE Trans. on Instrumentation and Measurement*, vol. 63, no. 4, pp. 877–888, April 2014.

[25] X. Gibert, V. M. Patel, D. Labate, and R. Chellappa, "Discrete shearlet transform on GPU with applications in anomaly detection and denoising," *EURASIP Journal on Advances in Signal Processing*, vol. 2014, no. 64, pp. 1–14, May 2014.

[26] R. Khan, S. Islam, and R. Biswas, "Automatic detection of defective rail anchors," in *IEEE 17th International Conference on Intelligent Transportation Systems (ITSC)*, Oct 2014, pp. 1583–1588.

[27] K. Fukushima, "Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position," *Biological Cybernetics*, vol. 36, no. 4, pp. 93–202, 1980.

[28] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel, "Backpropagation applied to handwritten zip code recognition," *Neural Computation*, vol. 1, no. 4, pp. 541–551, 1989.

[29] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, November 1998.

[30] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Systems (NIPS)*, 2013.

[31] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.

[32] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014.

[33] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell, "Caffe: Convolutional architecture for fast feature embedding," *arXiv:1408.5093*, 2014.

[34] R. Collobert, K. Kavukcuoglu, and C. Farabet, "Torch7: A matlab-like environment for machine learning," in *Advances in Neural Information Systems (NIPS)*, 2011.

[35] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.

[36] Y. Lv, Y. Duan, W. Kang, Z. Li, and F.-Y. Wang, "Traffic flow prediction with big data: A deep learning approach," *IEEE Transactions on Intelligent Transportation Systems*, vol. 16, no. 2, pp. 865–873, April 2015.

[37] W. Huang, G. Song, H. Hong, and K. Xie, "Deep architecture for traffic flow prediction: Deep belief networks with multitask learning," *IEEE Transactions on Intelligent Transportation Systems*, vol. 15, no. 5, pp. 2191–2201, Oct 2014.

[38] R. Caruana, "Multitask learning," *Machine Learning*, vol. 28, no. 1, pp. 41–75, Jul 1997.

[39] L. Y. Pratt, J. Mostow, and C. A. Kamm, "Direct transfer of learned information among neural networks," in *Proc. Of AAAI*, 1991.

[40] G. Hinton, "Learning distributed representation of concepts," in *Proc. of the 8th Int. Conf. of the Cognitive Science Society*, 1986, pp. 1–12.

[41] L. Fei-Fei, R. Fergus, and P. Perona, "One-shot learning of object categories," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 28, pp. 594–611, 2006.

[42] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov, "Improving neural networks by preventing co-adaptation of feature detectors," *arXiv preprint arXiv:1207.0580*, 2012.

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

[44] T. Ahonen, J. Matas, C. He, and M. Pietikäinen, "Rotation invariant image description with local binary pattern histogram fourier features," in *Image Analysis*. Springer, 2009, pp. 61–70.

[45] M. Muja and D. Lowe, "Fast approximate nearest neighbors with automatic algorithm configuration," in *International Conference on Computer Vision Theory and Application VISSAPP'09)*. INSTICC Press, 2009, pp. 331–340.

[46] T. Ojala, M. Pietikäinen, and T. Mäenpää, "Multiresolution gray-scale and rotation invariant texture classification with local binary patterns," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 24, no. 7, pp. 971–987, 2002.

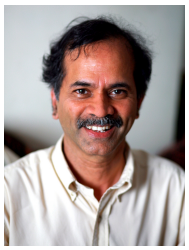
[47] M. Haghighat, S. Zonouz, and M. Abdel-Mottaleb, "Identification using encrypted biometrics," in *Computer Analysis of Images and Patterns*. Springer, 2013, pp. 440–448.

[48] B. Manjunath and W. Ma, "Texture features for browsing and retrieval of image data," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 18, no. 8, pp. 837–842, 1996.

[49] J. C. Platt, N. Cristianini, and J. Shawe-taylor, "Large margin DAGs for multiclass classification," in *Advances in Neural Information Systems (NIPS)*. MIT Press, 2000, pp. 547–553.

Xavier Gibert (M'03) received the B.S. degree in electrical engineering from Universitat Politècnica de Catalunya, Barcelona, Spain in 2002 and the M.S. and Ph.D. degrees in electrical engineering from the University of Maryland College, MD, USA, in 2004 and 2016, respectively. He is currently a Software Engineer at Google Robotics, Mountain View, CA. From 2004 to 2012, he was a scientist at ENSCO, Springfield, VA working on research and development of machine vision technologies for railway track inspection.

Vishal M. Patel (SM'01) received the B.S. degrees in electrical engineering and applied mathematics (Hons.) and the M.S. degree in applied mathematics from North Carolina State University, Raleigh, NC, USA, in 2004 and 2005, respectively, and the Ph.D. degree in electrical engineering from the University of Maryland College Park, MD, USA, in 2010. He is currently an Assistant Professor in the Department of Electrical and Computer Engineering (ECE) at Rutgers University. Prior to joining Rutgers University, he was a member of the research faculty with the University of Maryland's Institute for Advanced Computer Studies, College Park, MD, USA. His current research interests include signal processing, computer vision, and pattern recognition with applications in biometrics and imaging. He is a recipient of the 2016 ONR Young Investigator Award and the 2010 ORAU Post-Doctoral Fellowship. He is a member of Eta Kappa Nu, Pi Mu Epsilon, and Phi Beta Kappa.



Rama Chellappa (F'92) is a Minta Martin Professor of Engineering and Chair of the ECE department at the University of Maryland. Prof. Chellappa received the K.S. Fu Prize from the International Association of Pattern Recognition (IAPR). He is a recipient of the Society, Technical Achievement and Meritorious Service Awards from the IEEE Signal Processing Society and four IBM faculty Development Awards. He also received the Technical Achievement and Meritorious Service Awards from the IEEE Computer Society. At UMD, he received college and university level recognitions for research, teaching, innovation and mentoring of undergraduate students. In 2010, he was recognized as an Outstanding ECE by Purdue University. Prof. Chellappa served as the Editor-in-Chief of PAMI. He is a Golden Core Member of the IEEE Computer Society, served as a Distinguished Lecturer of the IEEE Signal Processing Society and as the President of IEEE Biometrics Council. He is a Fellow of IEEE, IAPR, OSA, AAAS, ACM and AAAI and holds four patents.