

# Integrated Multiple-Defect Detection and Evaluation of Rail Wheel Tread Images using Convolutional Neural Networks

Alexandre Trilla<sup>1,4</sup>, John Bob-Manuel<sup>2</sup>, Benjamin Lamoureux<sup>3</sup>, and Xavier Vilasis-Cardona<sup>4</sup>

<sup>1</sup> *Alstom, Santa Perpètua de la Mogoda, Barcelona, 08130, Spain*  
*alexandre.trilla@alstomgroup.com*

<sup>2</sup> *Alstom, Morden, London, SM45PT, United Kingdom*  
*john.bob-manuel@alstomgroup.com*

<sup>3</sup> *Alstom, Saint Ouen, Paris, 93482, France*  
*benjamin.lamoureux@alstomgroup.com*

<sup>4</sup> *DS4DS, La Salle, Universitat Ramon Llull, Barcelona, 08022, Spain*  
*xavier.vilasis@salle.url.edu*

## ABSTRACT

The wheel-rail interface is regarded as the most important factor for the dynamic behavior of a railway vehicle, affecting the safety of the service, the passenger comfort, and the life of the wheelset asset. The degradation of the wheels in contact with the rail is visibly manifest on their treads in the form of defects such as indentations, flats, cavities, etc. To guarantee a reliable rail service and maximize the availability of the rolling-stock assets, these defects need to be constantly and periodically monitored as their severity evolves. This inspection task is usually conducted manually at the fleet level and therefore it takes a lot of human resources. In order to add value to this maintenance activity, this article presents an automatic Deep Learning method to jointly detect and classify wheel tread defects based on smartphone pictures taken by the maintenance team. The architecture of this approach is based on a framework of Convolutional Neural Networks, which is applied to the different tasks of the diagnosis process including the location of the defect area within the image, the prediction of the defect size, and the identification of defect type. With this information determined, the maintenance-criteria rules can ultimately be applied to obtain the actionable results. The presented neural approach has been evaluated with a set of wheel defect pictures collected over the course of nearly two years, concluding that it can reliably automate the condition diagnosis of half of the current workload and thus reduce the lead time to take maintenance action, significantly reducing

engineering hours for verification and validation. Overall, this creates a platform of significant progress in automated predictive maintenance of rolling stock wheelsets.

## 1. INTRODUCTION

Wheel tread degradation is a common downtime cause for rolling-stock which can significantly affect service availability. Railway wheelsets are usually made of steel because of the high load they must bear and the generally high speed of this transport service. In this setting, it is in the wheel-rail interface that the incipient degradation damage develops as visible defects like cracks, spalls, shells, and skid flats (Magel, E., and Kalousek, J., 1996). If the severity of these defects compromises the safety operational considerations of the railway service (among other additional criteria, like the comfort of the passenger in high-speed rail), the trains are driven out of commercial service to perform a reprofiling maintenance action with the lathe in the depot. This activity is typically scheduled on a periodic mileage basis, but due to the nature of defect occurrence and its evolution, inspections are carried out as part of the regular maintenance procedure to guarantee the reliability of the service and extend the wheel life.

The inspections of wheel tread condition have been traditionally approached by monitoring dynamic variables (i.e., time-varying signals) such as the force and the strain, and also by using static variables like the raster image provided by a picture, which is rich in spatial information. And regarding the data-driven algorithms of their diagnosis methods, most of these strategies rely either on low-level/pixel-wise heuristics (Zhang, W., Zhang, Y., Li, J., Gao, X., and Wang, L., 2014;

Alexandre Trilla et al. This is an open-access article distributed under the terms of the Creative Commons Attribution 3.0 United States License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.  
<https://doi.org/10.36001/IJPHM.2021.v12i1.2906>

Hyde, P., Ulianov, C., and Defossez, F., 2016; Zhang, J., Guo, Z., Jiao, T., and Wang, M., 2018), or on maximum-margin classifiers like the Support Vector Machines (SVM) (Ma, K., Vicente, T. F. Y., Samaras, D., Petrucci, M., and Magnus, D. L., 2016; Guo, G., Peng, J., Yang, K., Xie, L., and Song, W., 2017). However, these solutions seem to be complementary, and neither clearly outstands its counterpart.

Out of the numerous endeavors to detect rail wheel defects, this work underlines the study developed by Krummenacher and colleagues, which compares an approach using wavelets with SVM to a time-series embedding with a Convolutional Neural Network (CNN), motivated by the recent success of this widely-adopted deep neural Computer Vision technology (Krummenacher, G., Ong, C. S., Koller, S., Kobayashi, S., and Buhmann, M., 2018). Their investigation concludes that the CNN approach improves the classification performance through its automatic representation learning ability. This result is much in line with the current popular Machine Learning (and in particular Deep Learning) research trend driven by CNN's ability to spot surface degradation problems (Han, K., Sun, M., Zhou, X., Zhang, G., Dang, H., and Liu, Z., 2017; Shang, L., Yang, Q., Wang, J., Li, S., and Lei, W., 2018; Zhang, Y., Cui, X., Liu, Y., and Yu, B., 2018).

Following state of the art Deep Learning techniques for Prognostics and Health Management (PHM) (Fink, O., Wang, Q., Svensén, M., Dersin, P., Lee, W.-J., and Ducoffe, M., 2020), the present work is concerned with the design and implementation of a rail wheel tread defect diagnosis system based on CNN applied to smartphone pictures that is able to cope with the increasing productivity demand to maintain more assets with the same resources and reduce the engineering lead time to take maintenance action (Vickerstaff, A., Bevan, A., and Boyacioglu, P., 2020). To attain this goal, this approach breaks down the complexity of the whole value chain into modules that may be developed in their own specific context, and it blends the hands-on experience available on the shop floor with the strong technical background available in the engineering office. In addition, an industrialized online web application based on modern software development tools and practices is also created to deploy this solution at the fleet level.

This article outlines the different steps involved in the development of this project: from the research that statistically states the feasibility of the proposed solution, to its industrialization through a minimum-viable product as a proof of concept. Section 2 describes the design procedure, including the description of the data, the learning technique and its evaluation, and the robust industrialized solution. Section 3 shows the expected performance results. Section 4 discusses the overall outcomes and the limitations of the approach, and Section 5 provides the conclusions of the work and reflects on its impact on the current maintenance plan along with the future avenues of improvement.

## 2. METHOD

This section describes process that has been followed to obtain a robust wheel tread defect diagnosis method.

### 2.1. Defect Data Description

To merge the knowledge from both the depot workshop and the engineering office, data from each environment needs to be available for learning. This section describes the kind of information that can be extracted from each perspective.

#### 2.1.1. Maintenance Data

A collection of 4600 wheel tread defect pictures taken with smartphones has been compiled over the course of two years by the maintenance repair and overhaul (MRO) team in the Alstom's Traincare Centre (i.e., the London Underground Northern Line fleet). The maintenance staff take pictures whenever an incipient defect is detected on the shop floor. The accumulated dataset depicts the presence of six different defects, which are described as follows with increasing severity:

**Indentation (INDT)** Superficial dent caused by the wheels running over a hard object on the track. This category also includes the "pitting" defect, which displays a similar effect on the wheel tread but its root cause is the mechanical strain.

**Rolling Contact Fatigue (RCF)** Cracks caused by repeated contact stress during the rolling motion of the wheels. RCF is a major wear issue in the London Underground infrastructure and its monitoring is incredibly labor intensive requiring precise visual inspection and detailed data recording (Vickerstaff, A., Bevan, A., and Boyacioglu, P., 2020).

**Wheel Flat (FLT)** Rash that appears on both wheels caused by the wheelset skidding on the rail.

**Clustering (CLUS)** Also known as multiple cavities, it has to do with the appearance of several bruises along the tread due to uneven contact issues.

**Spalling (SPALL)** Also known as single cavity or shelling, it is the critical development of one of the multiple cavities described before.

**Crazing (CRAZ)** Also known as thermal cracking, it is a fracture that occurs with repeated heating and cooling of the wheel tread surface caused by traction and braking actions.

As an example, Figure 1 shows a wheel tread picture with spall and RCF defects. In this dataset, though, there is a strong bias toward the RCF type (with over 80% of the instances). Such a major defect type imbalance may pose an adverse situation for Machine Learning (Yang, Y., and Xu, Z., 2020). Therefore, this work downsamples the RCF subset of data so that the resulting defect type distribution is more amenable to

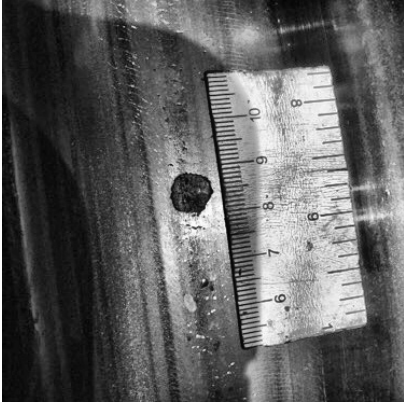


Figure 1. Wheel tread showing SPALL and RCF defects.

Table 1. Wheel tread defect dataset properties, including the number of instances, the defect type distribution, the contextual information (i.e., location and physical size of the defect), and the diagnosis assessment.

Attribute	MRO Data	ENG Data
Size	1200	118
INDT	23%	16%
RCF	36%	52%
FLT	21%	7%
CLUS	8%	11%
SPALL	11%	4%
CRAZ	1%	0%
(None)	0%	10%
Location	✓	✗
Physical size	✓	✗
Go	✗	15%
Warning	✗	51%
Stop	✗	34%

direct supervised learning. The reduced working maintenance dataset comprises 1200 picture instances, and its new defect type distribution is shown in Table 1. It can be seen that the crazing defect type is the underrepresented minority with only 1% of the instances. This skew is likely to cause some learning trouble, but that's an inherent difficulty in this environment that the proposed system will evaluate.

In addition to the graphical content of the picture, the maintenance staff also provides additional information in the form of textual data, identifying the inspected train unit, the physical size of the defect, etc. This unstructured context is processed with regular expressions to deal with the uneven spacing, the letter casing, etc., in order to complement the description of the spotted defects. Nevertheless, the dependability in this supplementary material may be questionable, and the picture remains to be the most reliable datum that the engineering team reviews for the definitive diagnostic. Therefore, the MRO context must only be used as an informative indication.

### 2.1.2. Engineering Data

In a similar vein, the engineering (ENG) team has curated a collection of 118 defects, see Table 1 for details. Note that this dataset is an order of magnitude smaller, and also exhibits a strong bias toward the RCF defect type. In addition, this set misses the “crazing” type, and it contains the absence of defect (i.e., images without a problem).

Although contextual data such as the location and the physical size of the defect are not available here, what is especially important is the condition assessment from the expertise, which also displays strong bias toward the “warning” statement. This is the engineering advice that drives the maintenance actions. In sight of the characteristics of the MRO and ENG datasets, which are both partially overlapping and complementary, there may exist some potential criteria transfer issues that need to be observed.

## 2.2. Image Processing

The collection of raster images that depict the wheel tread defects poses challenging issues due to the variability of the hand-held smartphone-based capture process. Depending on who is taking the picture and when, there is inconsistency in the focus, distance to the defect, lighting, etc. To address these concerns, a pixel-level Image Processing module is created.

### 2.2.1. Preprocessing

First, the three color channels (i.e., RGB) are conflated into one single intensity channel. The steel of the wheel treads is mostly blue-grayish, and any decoloration in the metal is equally visible with a shade on the resulting black-and-white picture, so the useful information is expected to be retained with this transformation. The image is now computationally lighter and therefore more tractable for further analysis.

Then, the edges of the picture, which may be taken vertically or horizontally, are trimmed so that the resulting image is standardized with a squared shape. Note that the area of interest containing the defect is always located around the center. With this operation, the size of the image is reduced to three quarters of its original size, which adds yet another time-computational advantage as less data needs be processed.

Finally, the histogram of the image is equalized to enhance its contrast (Pizer, S. M., Amburn, E. P., Austin, J. D., Cromartie, R., Geselowitz, A., Greer, T., Romeny, B. t. H., Zimmerman, J. B., and Zuiderveld, K., 1987). Figure 1 illustrates the application of these preprocessing steps to a defective wheel tread picture.

### 2.2.2. Data Augmentation

The abundance of data is required to design a Computer Vision solution based on Deep Learning, and the current defect

data collections are insufficient for use according to modern dataset size standards. In this situation, the system is likely to overfit and memorize the data, thus lacking the capacity to generalize. Therefore, a series of affine transformations (i.e., modifications that preserve the collinearity and the ratios of distances) are applied to these instances in order to augment their amount while retaining the salient degradation information (Simard, P. Y., Steinkraus, D., and Platt, J. C., 2003). Specifically, 4 translations (north, south, east, and west shifts), 2 rotations (clockwise and counterclockwise), and 4 mirrorings (horizontal, vertical, and the combined flipping) are performed. Additionally, 2 levels of additive white Gaussian noise are also applied. Eventually, the size of the dataset is increased 64-fold, yielding a working collection of over 80k instances (original and manufactured), which now enables exploring the data-driven solution. What is more, it is known that even small input perturbations like these are sufficient to considerably degrade the system's performance (Engstrom, L., Tran, B., Tsipras, D., Schmidt, L., and Madry, A., 2019). Therefore, by taking them into account during the training procedure, the final system is expected to increase its overall robustness against these potentially adverse effects (Hermann, K. L., Chen, T., and Kornblith, S., 2020).

### 2.3. Multitask System Architecture

To tackle the complexity of the wheel tread defect diagnosis problem, this work suggests a divide-and-conquer approach, where the main task is divided into five specialized data-driven modules:

**Defect Detection - Location (DD-Loc)** Identifies the central point of the defect area in the preprocessed image. This task is addressed as regression problem (i.e., landmark detection) where the coordinates of the defect location are predicted in pixel space.

**Defect Detection - Physical size (DD-Phy)** Predicts the size of the defect (width and length) in a given physical measure (e.g., millimeters). This task is also addressed as a regression problem.

**Defect Classification (DC)** Discriminates the different types of defects present in the defect area of the input picture. This task is addressed as a multi-label classification problem where the defects are not mutually exclusive, and the outputs represent defect membership probabilities. Ultimately, these probabilities are rated against a threshold  $\theta_{DC}$  and a discrete vector of potential defects is issued.

**Engineering Assessment (EA)** Determines the diagnostic based on the type of defect, its physical size, and a set of embedded logical rules that guarantee the minimum acceptance criteria. The output complies with a kind of traffic lights interface: go, warning, and stop.

**Confidence Index (CI)** Indicates the degree of trustworthiness in the provided diagnosis. Its output operates as a

binary variable.

Figure 2 shows the end to end diagnosis chain. Note that in addition to these five main data-driven modules, there is also the Image Processing (IP) block (already explained in Section 2.2), the defect cropping block, and the circumference calculation (CC) block. The latter two auxiliary blocks are self-explanatory.

### 2.4. Convolutional Neural Networks

The task division approach ensures that the multiple sources of learning signals do not get scrambled, so that each module can specialize. However, all these detection and classification problems operating on image data can be solved effectively with a convolutional neural architecture, mimicking the hierarchical feature learning strategy that occurs with the visual system's compositional structure (Bengio, Y., 2009), where the initial layers learn basic forms and the subsequent layers combine them to create complex patterns. CNN's are exceptionally successful at dealing with the high dimensionality of an image because they inherently reduce the solution search space (i.e., amount of learnable parameters) with a weight sharing strategy: they use a series of trainable filters that exploit the local surface statistical regularities of the pictures (Jo, J., and Bengio, Y., 2017), making the whole neural system less prone to overfit the data. In turn, this approach also makes these networks especially robust to location, detecting the same pattern in different parts of the photographs as the same filter kernel is reused throughout the image, which exhibits a translationally invariant structure.

Given all this common framework, this section describes a single flexible unified CNN to be applied to each task independently. For computational purposes there is an implicit image rescaling to 75 px that does not compromise the details of the defects, as the spatial aggregation of lower dimensional embeddings can be done without much or any loss in representational power (Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., and Wojna, Z., 2016).

#### 2.4.1. Framework Layout

Discovering neural network architectures remains a laborious but crucial task (Real, E., Moore, S., Selle, A., Saxena, S., Suematsu, Y. L., Tan, J., Le, Q. V., and Kurakin, A., 2017), because carefully balancing network depth, width, and resolution can lead to better performance (Tan, M., and Quoc, V. L., 2019). In the aim of taking advantage of the many years of focused investigation in neural layouts, the proposed CNN framework is fundamentally based on the classic LeNet-5 architecture (Lecun, Y., Bottou, L., Bengio, Y., and Haffner, P., 1998), which defines two convolutional stages and three fully connected stages, and the AlexNet architecture (Krizhevsky, A., Sutskever, I., and Hinton, G. E., 2012), which includes some Deep Learning improvements like the Rectified Lin-

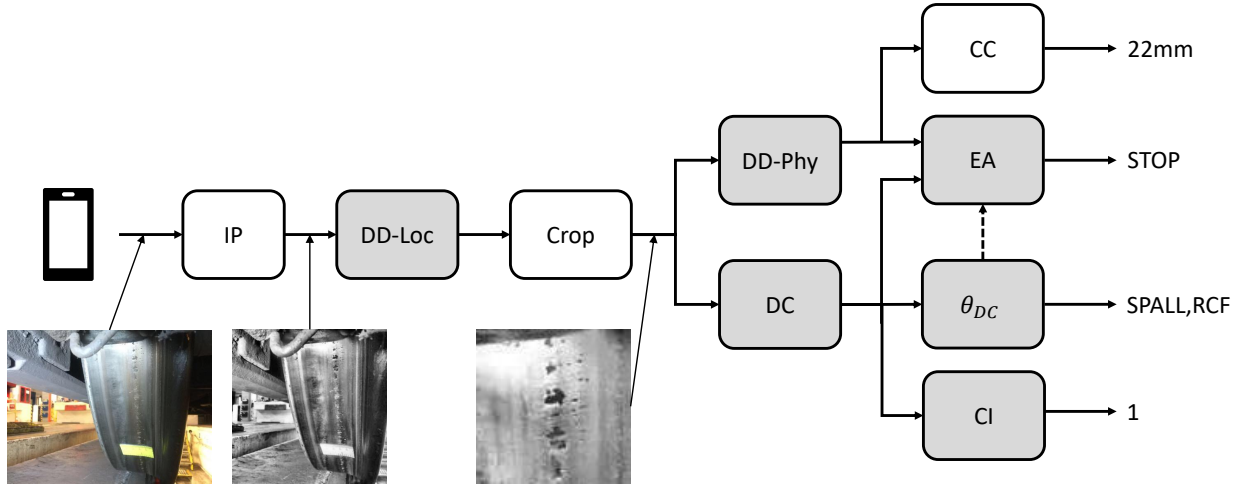


Figure 2. Wheel tread defect diagnosis framework. The main data-driven modules are: Defect Detection (DD-Loc and DD-Phy), Defect Classification (DC and  $\theta_{DC}$ ), Engineering Assessment (EA), and Confidence Index (CI). These are highlighted in shade. The auxiliary modules are: Image Processing (IP), Cropping, and the Circumference Calculation (CC). These are shown in white.

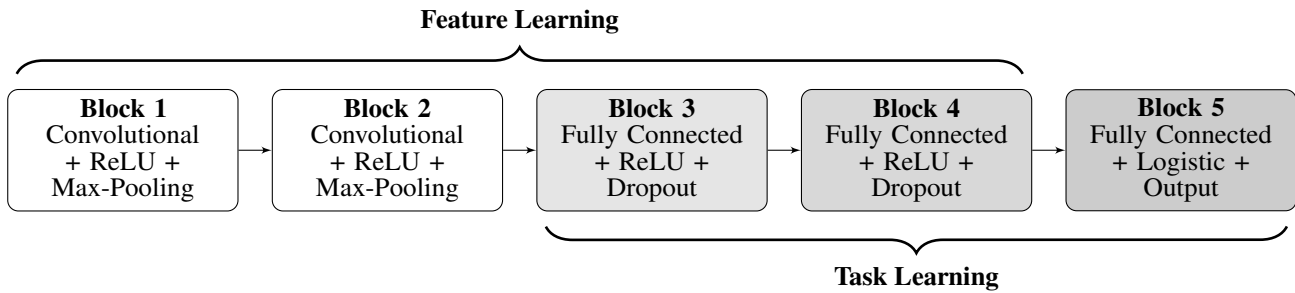


Figure 3. Functional blocks of the proposed versatile unified CNN, each of them containing a layer of learnable weights, an element-wise non-linearity with the ReLU activation function, and a layer of regularization. The Feature Learning blocks are displayed with a white background, whereas the Task Learning blocks have a light shade, showing the transition from the input data to the desired output result.

ear Unit (ReLU) as the non-linear activation function to train faster (Nair, V., and Hinton, G. E., 2010) and avoid the vanishing gradient problem (Glorot, X., Bordes, A., and Bengio, Y., 2011), and Dropout (i.e., random neuron deactivation) to preclude the co-adaptation of the feature detectors (Hinton, G. E., Srivastava, N., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. R., 2012) and prevent overfitting (Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R., 2014). Additionally, a subsampling overlap with a minimum stride of 1 px in a max-pooling step is considered to merge features, increase the robustness to noise, and improve the generalization. In summary, the basic building block of the proposed CNN combines a layer of adjustable weights like the convolutional filters or the fully dense connections, a non-linear rectification transformation (i.e., always positive neuron output values), and a layer of regularization with max-pooling or dropout. The idea of using a block of layers as a structural unit is gaining popularity (Khan, A., Sohail, A.,

Zahoora, U., and Qureshi, A. S., 2020), and therefore this approach is aligned with the latest trends in CNN architecture design. Figure 3 shows this layout, clearly identifying the two learning stages: the features and the task, which are described as follows.

#### 2.4.2. Feature Learning

The Feature Learning stage discovers the degradation-relevant traits in the pictures through a chain of non-linear convolutional and pooling operations, which initially learns simple shapes like curves and straight lines, and then combines these motifs to progressively create more complex and invariant compositions in a higher level of abstraction (Mahendran, A., and Vedaldi, A., 2015), just like many natural signals in visual neuroscience (LeCun, Y. and Bengio, Y., and Hinton, G. E., 2015). It is to note that in the proposed CNN design, no padding is used because there is no useful information in the borders of these images, which always display the defects

in the central region. Once the system has been trained, the adjusted weights of the initial layers (i.e., the image filters) may then be reused throughout the tasks (Donahue, J., Jia, Y., Vinyals, O., Hoffman, J., Zhang, N., Tzeng, E., and Darrell, T., 2013; Razavian, A. S., Azizpour, H., Sullivan, J., and Carlsson, S., 2014), which are learned in the following fully-connected layers. The next section delves into the details of this upcoming step.

### 2.4.3. Task Learning

The Task Learning step is acquired with the remaining fully-connected layer blocks that follow the convolutional blocks, see Figure 3. The non-linear learning capacity is guaranteed with this multilayer structure and the ReLU activation function. It is to note, though, that the last fully-connected block features a logistic sigmoid function, bounding the task dependent variable outputs between 0 and 1.

For the Defect Detection regression tasks (DD-Loc and DD-Phy), a maximum-value normalization step is performed taking into account the picture pixel size for the location task, and a reasonably large physical size for the measurement task. For these targets, a minimum squared error (MSE) training cost is used, which aims to reduce the real-valued prediction residuals.

For the Defect Classification task (DC), a binary class vector is used because the target degradation picture may have many labels (i.e., multiple defects on the same wheel tread). In this case, the cost function of use is the binary cross-entropy, so that each dimension of the output represents the posterior probability of the defect-class membership. This corresponds to the effective deployment of many logistic regressions following the one-vs-all multiclass strategy. Given that the defects are not mutually exclusive, the learning feedback will be shared among the intermediate layers. Finally, a heuristic decision rule based on a threshold is used to discretize the output: a defect class is selected if its predicted probability is over this minimum probability limit.

### 2.4.4. Feature/Task Embedding

This contribution states that the first two convolutional blocks are mainly meant to deal with the feature learning phase, and the three fully connected blocks that follow mostly learn the task at hand, see Figure 3 for the design diagram that shows the transition between the two stages. This feature/task integration is motivated by the local feature transfer aspect in the convolutional filters (Oquab, M., Bottou, L., Laptev, I., and Sivic, J., 2014), which can detect a particular pattern all over the picture, a characteristic that dense layers do not exhibit due to their rigidity. As it is, the proposed system learns a non-linear but rather shallow set of features, and a deep set of functional task operations. Nonetheless, the boundary between these two objectives in the network is not clear. The same

solution could have been equally described as a profoundly intricate feature learner with four blocks — two convolutional and two fully-connected — and a very shallow linear task learner with only one dense block, which is perhaps the generally adopted CNN functional interpretation. The obtained results would have been the same, especially if the different CNN's are freshly trained or the parameters are reused only for initialization pretraining purposes, but their interpretation would be different.

This work puts forward the contention that the task-specific learned knowledge is effectively embedded in the intermediate fully-connected hidden layers, as their large expressiveness supports this capacity (over 8 million tunable weights for this approach), see Table 2 for a detailed description of the system parameters. Although it has been pointed out that the hidden units may learn similar representations that converge to analogous features across the tasks (Kornblith, S., Norouzi, M., Lee, H., and Hinton, G., 2019), these layers may also experience some optimization difficulties (Yosinski, J., Clune, J., Bengio, Y., and Lipson, H., 2014) (i.e., layers FC3 and FC4). In this last cited reference it is documented that the transferability of features decreases as the distance between the base task and target task increases, thus supporting the rigid task-specific learned knowledge, and limiting the extent of their parameter reuse. This work suggests that only the first two convolutional blocks may be inherited in a different task and all the intermediate dense layers are to be retrained for each different objective.

## 2.5. Performance Evaluation

Different key performance indicators are used to evaluate the operation of the task-driven CNN approaches. The regression objectives are assessed with the variability of the resulting error distribution for a given confidence interval. This figure is indicative of the amount of epistemic uncertainty. For the classification task, the overall system performance is obtained with the macro-averaged accuracy, precision, and recall metrics (Duda, R. O., Hart, P. E., and Stork, D. G., 2001). These values represent the rate of good classifications, and the penalties that false alarms and missed defects introduce.

In the scenarios where the same dataset is used for learning and evaluation, the performance values are generally estimated with Monte Carlo cross-validation (Dubitzky, W., Granzow, M., and Berrar, D., 2007). Specifically, four rounds of repeated random subsampling are applied on a stratified set of defect types with a train/test split rate of 80/20 (%), which should yields an error sample size over 1k instances that is sufficient to reliably conduct the statistical calculations. In the scenarios where the working dataset is too small for applying this approach, then a leave-one-out cross-validation strategy is pursued. Finally, in the scenarios where both datasets are used, the MRO data is used for training, and the ENG data is

Table 2. CNN parameter chart. The Dropout layers feature a probability of 0.1, and the OR or OC represent the regression or the classification output.

Block	Layer ID	Type	Filter	Stride	Amount	Units	Activation	Parameters
1	C1	Conv2D	(5,5,1)	(1,1)	6	(71,71,6)	ReLU	156
1	P1	Max Pool	(2,2)	(1,1)		(70,70,6)	Linear	0
2	C2	Conv2D	(5,5,6)	(1,1)	16	(66,66,16)	ReLU	2416
2	P2	Max Pool	(2,2)	(1,1)		(65,65,16)	Linear	0
3	FC3	Dense				120	ReLU	8112120
3	D3	Dropout				120	Linear	0
4	FC4	Dense				84	ReLU	10164
4	D4	Dropout				84	Linear	0
5	OR	Dense				2	Logistic	170
5	OC	Dense				6	Logistic	510

held out for testing.

## 2.6. Development and Industrialization

The Machine Learning research is entirely conducted with the Python3 programming language and its data science ecosystem environment for PHM (Rezaeianjouybari & Shang, 2020), mainly led by NumPy, Scikit-learn and SciPy. For the image processing tasks, OpenCV and scikit-image are also used. Finally, the intensive computations that Deep Learning entails are carried out by TensorFlow2 (Guo, Q., Chen, S., Xie, X., Ma, L., Hu, Q., Liu, H., Liu, Y., Zhao, J., and Li, X., 2019).

The industrialization of the proposed solution for creating a minimum-viable product leverages the latest developments of the open-source big data ecosystem (Cui, Y., Kara, S., and Chan, K. C., 2020). The full architecture stack is running on top of a cluster of machines managed by Kubernetes, a well-proven system to automate, scale and ensure high availability of computer applications. Kubernetes has been increasingly used in the field of machine learning over the past years (Aji, I. P., and Kusuma, G. P., 2020; Wu, C., Haihong, E., and Song, M., 2020). It is divided into four layers: (A) the data layer stores all the data used by the product; (B) the flow layer orchestrates and schedules the “hand-to-hand” transfer of data between the different applications; (C) the application layer centralizes all the “business-value” functions performed by the wheel tread defect diagnosis framework presented in this paper, and (D) the presentation layer contains the user app. The technologies used for each layer, illustrated in Figure 4, are described as follows:

**Data Layer** PostgreSQL (Juba, S., and Volkov, A., 2019) is used to store the application data such as users, passwords and computation results. It is a well-proven tool with a very powerful query engine. It is used jointly with PostgREST application that creates a REST API on top of PostgreSQL and avoids direct connections which are risky in terms of cybersecurity. MinIO cloud storage (Johnston, C., 2020) is used to upload, store and download the images. It is based on Amazon S3 technology which is

able to handle multiple large binary files downloads and uploads simultaneously without any loss of performance.

**Flow Layer** Apache NiFi (Chanthakit, S., Keeratiwintakorn, P., and Rattanapoka, C., 2019) is used to orchestrate back and forth the delivery of data between the application and the data layers. It provides a very user-friendly web interface with multiple types of functional blocks (so-called processors) that one can organize and connect together to create more complex flows. One can then follow the traces of the processing path directly in the web interface, which is very practical to monitor the progress. The underlying Kubernetes allows NiFi to run a single flow in a cluster of multiple machines at the same time, thus ensuring the availability of the product.

**Application Layer** OpenFaaS (Balla, D., Maliosz, M., and Simon, C., 2020) is used to expose the Python scripts for the wheel tread defect diagnosis as a web service executable through a HTTPS request. First the Python scripts and models are encapsulated into a Docker image that is pushed to the OpenFaaS registry. Then OpenFaaS manages the deployment of the Docker image and the routing of requests. OpenFaaS is also increasingly used in the field of Machine Learning (Jang, R-Y., Lee, R., Park, M.-W., and Lee, S.-H., 2020). The underlying Kubernetes allows OpenFaaS to automatically scale up the number of deployed Docker images to smartly adapt the computational power to the actual quantity of requests.

**Presentation Layer** The Ionic (Yusuf, S., 2016) software development kit is used to develop the user app. The user interface is built as a Progressive Web App (PWA) using the Angular framework jointly with web technologies such as CSS and HTML5. The use of PWA technology allows the mobile app to run both on mobile and web devices (Bjørn-Hansen, A., Majchrzak, T. A., and Grønli, T.-M., 2017). The app communicates through classical HTTPS GET and POST requests: with MinIO to post the images and with PostgREST API to get app parameters and computation results.

The main use-case scenario, presented Figure 4, is the follow-

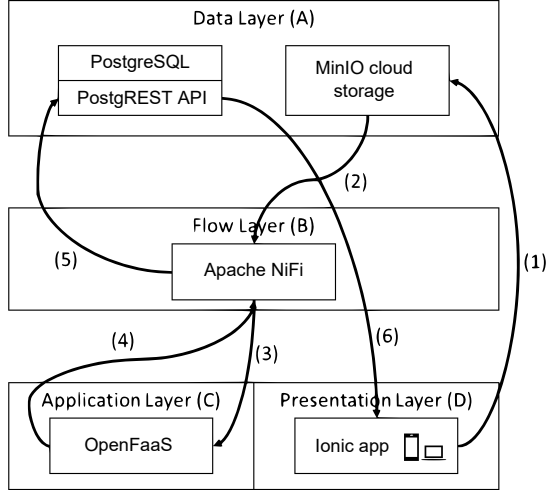


Figure 4. Industrialized architecture of the wheel tread diagnosis framework: four-layer stack with main use-case scenario.

ing: (1) a new image is posted by the maintainer from the user app to MinIO, (2) NiFi takes the image from MinIO, (3) NiFi posts the image to the OpenFaaS gateway to execute the wheel tread defect diagnosis function on its content, (4) OpenFaaS responds to the request with the results of the computation, (5) NiFi inserts the results into the PostgreSQL database through the PostgREST API, and (6) computation results are retrieved by the app and presented to the maintainer (and to the engineer) in the user interface according to the usage profile. The impact of these results on the maintenance business are presented in the following section.

### 3. RESULTS

This section details the results of the proposed CNN approach to the different specialized tasks to diagnose wheel tread defects and estimates their expected performance.

#### 3.1. Defect Location Performance (DD-Loc)

The defect location module is developed with the MRO dataset. Figure 5 shows the location prediction error distribution scored as the difference between the X and Y coordinates indistinctly. This result shows that the prediction error is centered around the target because there is no bias toward the left/right or up/down. The uncertainty is of 9.5 px, which corresponds to 12.66% of the image size.

#### 3.2. Physical Size Performance (DD-Phy)

The physical size prediction module is also developed with the MRO dataset. Figure 6 shows the error distribution scored as the difference between the width and the height indistinctly. This result shows that the error is sharply centered around the target. The uncertainty of the prediction is of 6.2 mm.

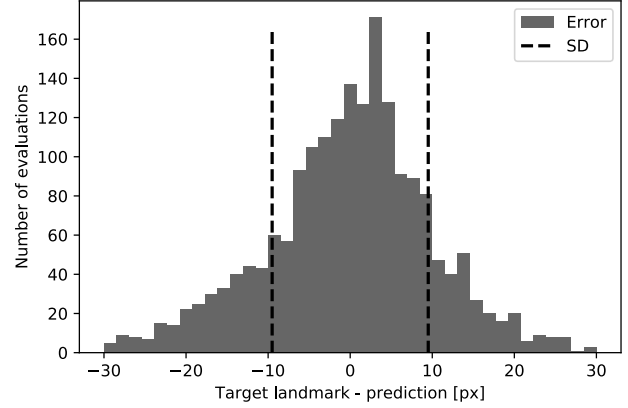


Figure 5. Histogram of the defect location prediction error. The 68% confidence interval SD (i.e., 1 standard deviation under the normality assumption) indicates the uncertainty.

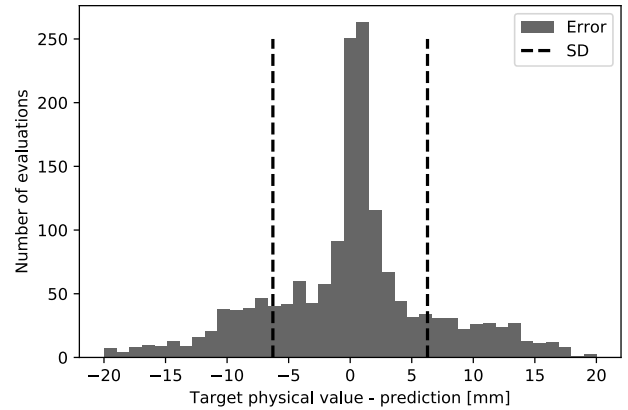


Figure 6. Histogram of the physical size prediction error. The 68% confidence interval SD (i.e., 1 standard deviation under the normality assumption) indicates the uncertainty.

#### 3.3. Defect Classification Performance (DC, $\theta_{DC}$ )

The classification module that scores the defect type membership probabilities (DC) is trained with the MRO dataset, and the threshold module that discretizes the result ( $\theta_{DC}$ ) is adjusted with the ENG dataset. Figure 7 shows the resulting classification metrics. Note that two potential work points can be identified in the diagram. Their characteristics are described as follows:

- Conservative work point (CWP,  $\theta_{DC} = 0.35$ ): minimize false negatives. With a lower threshold the system yields many potential failure candidates so that the risk of missing a problem is kept low, which is especially important from a safety perspective. The accuracy is higher (0.75) for this configuration.
- Eager work point (EWP,  $\theta_{DC} = 0.7$ ): minimize false alarms. With a higher threshold the system yields fewer potential failure candidates so the system increases its precision (around 0.3). In this configuration, the system is



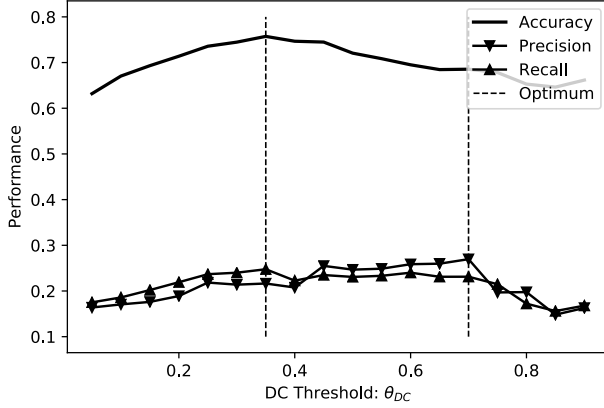


Figure 7. Macro-averaged Defect Classification metrics: accuracy, precision, and recall. Results are shown along with the probability discretization threshold  $\theta_{DC}$ . Two potential work points are identified.

sues “defect absence” labels whenever all the defect-type probabilities are low, thus enabling it to detect anomalies in compliance with the ISO 13374 international standard (ISO, 2003), i.e., operating as a dichotomic function.

Figure 7 is a kind of Receiver-Operating Characteristic curve, showing more than two key performance indicators. Note that if the threshold  $\theta_{DC}$  that operates on the vector of defect type probabilities is raised even further (beyond the Eager Work Point), the system is unable to raise any alarm as the required minimum probability values get close to 1.0, and therefore the precision and recall classification metrics drop because they both depend directly on the True Positives of the confusion matrix. Their expected “inverse” behavior is clearly observed at  $\theta_{DC} = 0.45$ , when the two curves cross. At that point, the system weighs equally the effect of False Positives and False Negatives. In terms of business impact, the priority criteria of the customer ultimately lead the performance tuning process.

Also note that the accuracy performance indicator is not reliable in this imbalanced data scenario, as the system might be biased toward the majority defect type (i.e., RCF), so further operational context is necessary for the evaluation. In the next section, these additional criteria are considered to give a better view of the actual expectations that this proposal provides.

### 3.4. Engineering Assessment Performance (EA)

This is probably the most decisive module of the system because it provides the actionable feedback in the form of “go - warning - stop” label statements. It is a purely task learning block developed with the ENG dataset. It is built with two of the fully-connected layers of the CNN, yielding a multilayer perceptron architecture. The resulting hidden embedding is arbitrarily set to 10 units (slightly greater than the input dimensionality built with the outputs of the former modules) with Dropout, which will prevent overfitting and ensure that

Table 3. Engineering Assessment performance focused on potential SAF according to different work scenarios: MAC logic rules and conservative/eager work points (CWP/EWP).

Probability	No MAC	MAC + CWP	MAC + EWP
$p(SAF stop)$	0.5	0.34	0.36
$p(SAF warn)$	0.32	0.4	0.32
$p(stop)$	0.08	0.96	0.47
$p(warn)$	0.92	0.04	0.53
$p(SAF)$	0.33	0.34	0.34
$p(SAF; ENG)$	0.37	0.98	0.64

the network automatically finds its optimum expressiveness. In addition, this EA module may eventually apply a series of logical rules known as the minimum acceptance criteria (MAC), which are conservative in nature, to guarantee that certain critical limits are never exceeded.

For the design of this module, its performance in the following three configurations is taken into consideration: no MAC rules, MAC rules with the conservative work point, and MAC rules with the eager work point. Table 3 shows the performance results in probabilistic terms derived from the confusion matrices, and focusing on the potential service-affecting failures (SAF), which are the critical situations identified by the engineering office (i.e., a “stop” label in the ground truth).

This analysis clearly shows the different operating modes: the purely data-driven scenario (i.e., no MAC) is strongly biased toward issuing warning results (just like the majority of the dataset), the conservative scenario is strongly biased toward raising alarms, and the eager scenario is balanced. However, the probability of actually detecting the SAF, which is calculated with the law of total probability, see Eq. (1), is almost the same in all scenarios. Note that the system does not report any “go” result, which may be reasonable because the maintenance staff only take pictures if they suspect the presence of an incipient defect.

$$p(SAF) = \sum p(SAF|diagnosis) \cdot p(diagnosis) \\ p(SAF|stop) \cdot p(stop) + (SAF|warn) \cdot p(warn) \quad (1)$$

In the light of this inconclusive result where all the approaches yield a probability around 0.34 to detect the potential SAF, the contribution of the engineering team will be determining to break the tie.

#### 3.4.1. Engineering Verification and Validation

Whenever the engineering team checks a picture, it always detects the potential SAF. At present, the engineers manually review the whole stream of images, which takes a lot of person-hour resources and this workload may hinder the completion of other activities. To add value to the overall maintenance pro-

cedure reducing the weight of this engineering validation task, this work proposes that only the pictures automatically rated with the “stop” diagnosis are to be manually checked by the engineering office. In consequence,  $p(SAF|stop; ENG) = 1$ , and the probability of detecting the SAF increases in different degrees according to the given design strategy. The bottom row of Table 3 shows the impact of this new criterion. These results indicate that with the eager approach (along with the MAC rules) the engineering team can reduce its current workload more than 50%, and retain a SAF detection rate of 64%. This is regarded as the optimum trade-off between the complete manual workload and the complete automated approach, potentially resulting in the best pay off for the adoption of the proposed system.

### 3.5. Confidence Index Performance (CI)

The Confidence Index informs that the system is self aware of the reliability of its predictions. This indicator is developed with the ENG dataset through a heuristic function that determines the result of this quality test. This function operates on the vector of probabilities of the preceding DC module, and applies an Active Learning approach known as an “acquisition function” that determines the degree of uncertainty in the classification (for all the defect types  $D$ ) through its entropy (Settles, B., 2010). The resulting value is finally scored against a maximum threshold  $\theta_{CI}$  to obtain the binary-valued CI, shown in Eq. (2).

$$CI = \left( - \sum_{\forall d \in D} p(d) \cdot \log(p(d)) \right) < \theta_{CI} \quad (2)$$

Figure 8 displays the distribution of the DC entropy for the ENG data, related to their diagnosis labels. Note that for all the instances that display an entropy lower than  $\theta_{CI} = 1.2$ , the rate of the “stop” diagnostic (i.e., the potential SAF) over the other labels in each bin is considerably greater than the rate over the entropy value of 1.2. Thus, this leads to the conclusion that  $\theta_{CI} = 1.2$  is the adequate threshold for the Confidence Index.

The engineering office capitalizes the maintenance expertise, understands the limitations of the proposed CNN system, and the CI indicator can be used to drive their decisions, among other functional criteria. The following section is dedicated to this latter point.

## 4. DISCUSSION

This section elaborates upon the contextual behavior of the wheel tread defect diagnosis system described in this work.

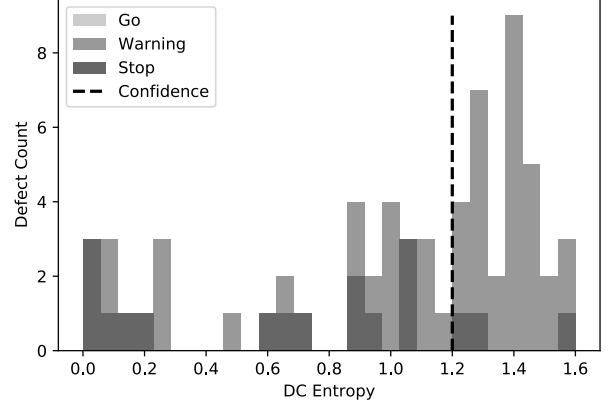


Figure 8. Histogram of the DC entropy for the ENG data with respect to their diagnosis labels.

### 4.1. Understanding the Learned CNN System

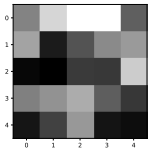
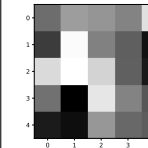
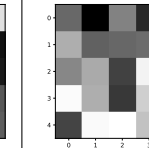
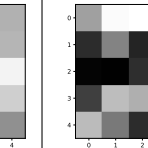
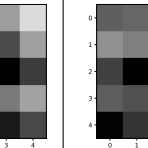
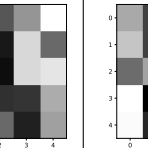
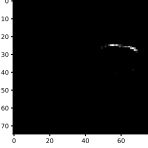
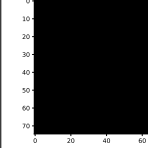
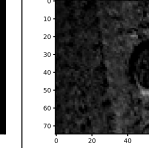
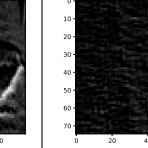
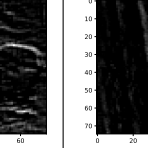
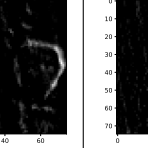
In this multi-label setting where many defects may be represented in the same image, learning the templates for arrangements of objects becomes rapidly intractable because of the combinatorial explosion in the number of features to be stored (Ricci, M., Kim, J., and Serre, T., 2018). One of the main criticisms generally attributed to neural networks, and thus to CNN’s in particular, is their lack of interpretability or explainability, what is also known as a “black box” interface behavior (Fong, R., and Vedaldi, A., 2017). The following two sections delve into the internal working details of the learned CNN system in order to shed some light into their cumbersome operations, revealing the desirable properties of compositionality and class discrimination that CNN’s are expected to exhibit (Zeiler, M. D., and Fergus, R., 2014).

#### 4.1.1. Image Filters

A CNN is fundamentally characterized by the adapted design of its filters, which get convoluted with the input image in order to highlight interesting patterns, just like the human visual system (Eickenberg, M., Gramfort, A., Varoquaux, G., and Thirion, B., 2017). In a sense, these filters are like templates that match specific motifs in the pictures, especially the ones found in the first layer of a vision system (Erhan, D., Bengio, Y., Courville, A., and Vincent, P., 2009), where the receptive field, i.e., the size of the region in the input that produces the feature, is minimum and corresponds to the size of the filter (Le, H., and Borji, A., 2017). It is widely accepted that these first functions learn edge-detecting Gabor filters, i.e., linear functions used for texture analysis that highlight a specific frequency content in a specific selective direction. Therefore, analyzing them at the pixel level reveals relevant information about the captured knowledge (Bach, S., Binder, A., Montavon, G., Klauschen, F., Müller, K.-R., and Samek, W., 2015).

The outputs of the filters correspond to specific locations of

Table 4. First layer of learned image filters (C1) and their impact on a sample image showing two defects (SPALL and RCF).

<b>Filter</b>						
<b>Output</b>						
<b>Function</b>	Up curve	(None)	Down curve	Up/down curve	Vertical line, right curve	Vertical line, right/left curve
<b>Defect</b>	SPALL	(None)	SPALL	SPALL	SPALL, RCF	SPALL, RCF

interest whenever their activation is high, thus creating a spatial feature detector (Zeiler, M. D., and Fergus, R., 2014). And given that these patterns can be observed in any place around the picture, their dependence on individual units is reduced, thereby improving the network generalization performance (Morcos, A. S., Barrett, D. G., Rabinowitz, N. C., and Botvinick, M., 2018). Table 4 shows the first filters that the system has learned (i.e., layer C1) and the impact of their design on a sample image that contains two tread defects (SPALL and RCF). As it can be seen, each of the six input filters learns a particular detail of the degradation: some filters learn curves, others learn straight lines, and even two of them learn both features, illustrating the multifaceted character of the related neurons (Nguyen, A., Yosinski, J., and Clune, J., 2016). In most cases, their output can then be directly related to a specific type of defect, which gives them a kind of latent representation aligned with human-interpretable semantic concepts (Bau, D., Zhou, B., Khosla, A., Oliva, A., and Torralba, A., 2017). However, it is the entire space of activations, rather than the individual units, that contains the bulk of the semantic information (Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., and Fergus, R., 2013). Finally, all these features get blended into the layers that follow to accomplish some task-driven goal.

#### 4.1.2. Defect Manifold

This section evaluates the separability of the spatial distribution of the defects in the latent space (Chen, Z., Bei, Y., and Rudin, C., 2020). To see how the CNN architecture internally discriminates the data and manages the inter-defect knowledge (Mahendran, A., and Vedaldi, A., 2015; Simonyan, K., Vedaldi, A., and Zisserman, A., 2013; Zhou, B., Khosla, A., Lapedriza, A., Oliva, A., and Torralba, A., 2015) as well as the intra-defect knowledge through a hierarchical and compositional pipeline (Wei, D., Zhou, B., Torralba, A., and Freeman, W. T., 2015), Figure 9 shows the scattering of the instances on

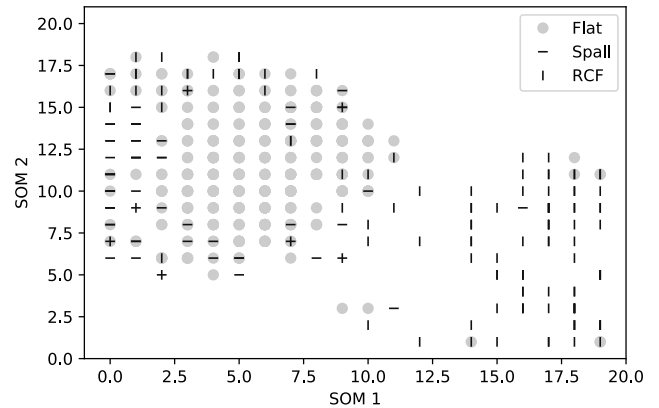


Figure 9. Self Organizing Map of the FC4 layer embedding for the defect classification task showing the main defect types: flat, spall, and RCF.

the penultimate adjustable layer FC4 for the defect classification task using a Self Organizing Map (SOM) (Kohonen, T., 1990). The SOM is an unsupervised non-linear transformation technique based on competitive learning that produces a discretized representation of the data preserving its topological properties, i.e., its similarity clusters. In PHM it has been used for anomaly detection and fault location purposes (Tian, J., Azarian, M. H., and Pecht, M., 2014; Zhao, W., Siegel, D., Lee, J., and Su, L., 2013), also in railway systems (Alessi, A., La-Cascia, P., Lamoureux, B., Pugnali, M., and Dersin, P., 2016).

In the scenario presented in this work, the SOM shows how the CNN learns to separate the three major defect prototypes: RCF, flat, and spall. In particular, it can be observed that the system learns to differentiate straight-line patterns (e.g., RCF), which are clustered to the right, from rounded patterns (i.e., spall and flat), which are clustered to the left. In this latter categorization, the overlap illustrates that the curvy-type

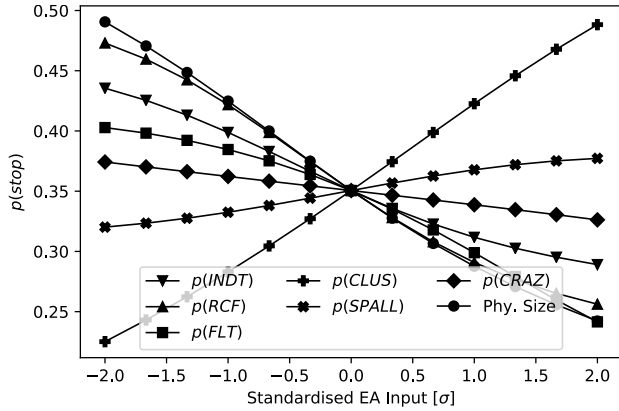


Figure 10. Engineering Assessment sensitivity analysis through the profile method for the “stop” diagnosis statement. The inputs have been standardized.

patterns seem to follow a hierarchical structure over the defects (Alsallakh, B., Jourabloo, A., Ye, M., Liu, X., and Ren, L., 2018). Note that only the last layer of the CNN deals with this manifold representation, and the eventual classification thus needs to be attained with linear discriminators, which seems to be adequate based on the lower-dimensional representation provided by the SOM. However, a proper expressiveness analysis with an additional hidden layer, thus creating a multi-layer perceptron, could be a more general solution (Simard, P. Y., Steinkraus, D., and Platt, J. C., 2003), following the universal approximation theorem for neural networks (Cybenko, G., 1989; Pinkus, A., 1999).

#### 4.2. Engineering Assessment Sensitivity

The Engineering Assessment module is arguably the most critical point in the system because it provides the actionable feedback to the maintainer. To understand its inner working mechanism through the impact of the input variables (i.e., the defect type probabilities and the physical size of the defect) on the output diagnosis, Figure 10 displays the result of a sensitivity analysis based on the profile method (Shojaeefard, M. H., Akbari, M. Tahani, M., and Farhani, F., 2013) for the critical “stop” diagnostic.

Assuming that the importance of a variable is driven by the dynamic range of the output, it is shown that the physical size, the RCF, and the CLUS probabilities lead this ranking. In addition, the physical size and the RCF probability variables are strongly negatively correlated with the “stop” probability diagnosis. The RCF is a type of defect that by itself does not directly halt the railway service, so this negative relationship makes sense. The physical defect, however, does not reasonably follow this criterion, but its impact is highly correlated with the RCF and this is what the EA module has ultimately learned. Finally, the CLUS probability is strongly positively correlated with the diagnosis, which makes perfect sense be-

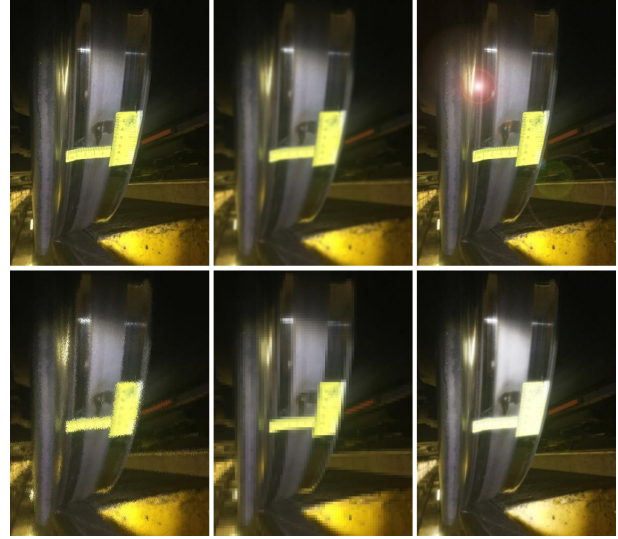


Figure 11. Examples of image modifications: normal picture, blur, glare, noise, pixelation, and shine.

cause this is a critical defect type.

#### 4.3. Robustness to Feature Corruption

Machine learning solutions, including neural networks and Deep Learning, may exhibit unexpected instability on simple perturbations. Therefore, they are at risk of being tricked by adversarial instances, which are intentionally corrupted data that lead the system to output incorrect results with high confidence (Goodfellow, I. J., Shlens, J., and Szegedy, C., 2015). Moreover, image processing applications are especially targeted by these attacks because some of these small perturbations are difficult to detect as they exploit edge cases. Methods such as histogram equalization, see Section 2.2.1, are helpful to prevent them (Hendrycks, D., and Dietterich, T., 2019), but careful attention is needed because cybersecurity in railways is an area that has attracted a lot of interest recently due to an increasing number of denial-of-service attacks (Masson, É., and Gransart, C., 2017).

A useful approach to build a defense against these adversarial attacks is to construct a predictor that is robust to the deletion of features at test time (Globerson & Roweis, 2009). In this sense, the Engineering Assessment module already features a Dropout layer after the embedding, see Section 3.4. In the defect diagnosis scenario based on smartphone pictures presented in this work, the proposed system should be robust to artificial image modifications that could be used in an adversarial attack, including effects like blurring, flash glare, etc. Figure 11 shows some typical examples of these kind of tweaks, and Table 5 evaluates their impact on the final diagnosis.

This analysis of feature perturbations shows that the proposed system exhibits a fairly good overall robustness to potential im-

Table 5. Image modifications and their impact to the proposed defect diagnosis system.

Filter/Effect	Size (mm)	Defects	Diagnosis	CI
Normal	55	FLT	Stop	1
Blur	54	None	Warning	0
Glare	61	FLT	Stop	1
Noise	53	None	Warning	1
Pixelation	68	FLT	Stop	0
Shine	53	FLT	Stop	1

age corruptions, including the common shine and glare effects produced by the flash. Robustness to pixelation also indicates that the resolution of the smartphone camera is sufficient. Nevertheless, the blur and noise perturbations cause the system to fail, as a warning signal is issued instead of the expected “stop” statement. These situations thus need to be avoided through the recommendation of taking still photographs in a dust-free environment.

#### 4.4. Pragmatic Project Management

The development of an industrial Deep Learning solution entails having to deal with many different components, and this leaves the door open to many different potential approaches. On the data acquisition stage, a Computer Vision engineer will probably argue that the system improvement lies on the quality of the pictures, and these smartphone images do have focus issues, uneven lighting conditions, different distances to the wheelset defect of interest, etc. However, when the input pictures are taken at different scales, the CNN will extract features at different scales (He, K., Zhang, X., Ren, S., and Sun, J., 2015), so these variations should not be the primary point of concern. Moreover, it has been shown that the resolution of the camera (leading to a pixelation effect) does not critically impact the diagnosis.

What has been observed is the tight dependence on labeled data to develop such a system. The annotation process is tedious, and fatigue builds up after some time. In this work, a standalone computer application has been developed to iterate the dataset and record the expertise, which has been provided by one single expert per instance. A minimum inter-annotator agreement rate is not strictly necessary for a feasible tagging of maintenance data (Hastings, E. M., Sexton, T., Brundage, M. P., and Hodkiewicz, M., 2019). However, further progress in this line should be provided by unsupervised or semi-supervised approaches, which reduce the amount of repetitive human effort (Bengio, Y., 2009), like the Meta Pseudo Labels approach (Pham, H., Dai, Z., Xie, Q., Luong, M.-T., and Le, Q. V., 2020), where a teacher network is trained to generate pseudo labels on unlabeled data to train a student network, and adapts with the performance of the student network on the labeled dataset. Additionally, a strategy to reduce the bias in the data (Kim, B., Kim, H., Kim, K., Kim, S., and Kim, J., 2019) and the noise in the labels (Lee, K.-H., He, X.,

Zhang, L., and Yang, L., 2018) should also be explored.

On the learning stage, the proposed CNN design displays zero bias error throughout the different modules, and any tweak beyond this neural design has led to the appearance of some average loss (keeping the same uncertainty). Therefore, as it is, the described approach shows an optimum complexity for this defect diagnosis problem, despite the obtained results are far from perfect. However, it is not clear how a different architecture might be of help in this scenario. There are some approaches that suggest using smaller convolutional filters (3x3) along with a network depth of 16 to 19 layers (Simonyan, K., and Zisserman, A., 2015), keeping a constant computational budget for the industrialization (Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., and Rabinovich, A., 2015), using kernels within the convolution function (Ammann, O., Michau, G., and Fink, O., 2020), or dropping the pooling layers due to their seldom attributed destructive role (Springenberg, J. T., Dosovitskiy, A., Brox, T., and Riedmiller, M., 2015). The Deep Learning field is in full blossom at present, and potentially many different solution approaches to the problem will be developed, so further research is required to get an optimal solution and to settle into the plateau of general productivity. Ultimately, the obtained solution as it is could be used to train a new generation of networks in a self-distillation manner and push the test performance a bit further (Zhang, L., Song, J., Gao, A., Chen, J., Bao, J., and Ma, K., 2019).

Alternatively, the current technology may also be used with a different perspective: instead of the proposed modular approach, a truly multitask environment could also be explored, because a single network can manage to do classification and regression tasks concurrently (Zhou, B., Khosla, A., Lapedriza, A., Oliva, A., and Torralba, A., 2015; Sermanet, P., Eigen, D., Zhang, X., Mathieu, M., Fergus, R., and LeCun, Y., 2013). What is more, the application of a CNN at multiple locations in a sliding window fashion (instead of the full image input) has also been reported to be successful (Sermanet, P., Eigen, D., Zhang, X., Mathieu, M., Fergus, R., and LeCun, Y., 2013; Oquab, M., Bottou, L., Laptev, I., and Sivic, J., 2014). In addition, the domain transfer between a rich image environment like ImageNet and the defect problem at hand may also be of help to learn better feature representations and improve the system generalization (Salman, H., Ilyas, A., Engstrom, L., Kapoor, A., and Madry, A., 2020; Kornblith, S., Shlens, J., and Le, Q. V., 2019). Furthermore, the consideration of synthetic data including adversarial images (Ilyas, A., Santurkar, S., Tsipras, D., Engstrom, L., Tran, B., and Madry, A., 2019), which is a popular approach to train Deep Learning models for Computer Vision (Nikolenko, S. I., 2019), is a useful resource to enhance the robustness of the system. And in the line of continuous improvement, if the user feedback is included with respect to the presented diagnosis results, the system can also exhibit some sort of enhanced evolution as

new data is processed.

Finally, regarding the industrialization, the main limitation of the proposed architecture is that the Python models used for the whole machine learning process are included in a Docker container image that is almost immutable, hence dynamic updates of the models are cumbersome. An upgrade to the proposed solution could include a kind of model registry that is periodically called to download new versions of the models, which would be developed by the data scientist in the loop following a continuous improvement procedure driven by the Return on Experience of the product, including new features, bug fixes, patches motivated by incorrect predictions, etc.

## 5. CONCLUSION

The detection of railway wheel tread defects on raster picture data is a daunting task that involves many different levels of analysis. This paper presents an integrated solution based on many Convolutional Neural Networks that locate the damaged areas in the images, estimate the physical size of the shown defects, and assess their type and severity. This proposal describes a task-division approach that helps understand the caveats and pitfalls of the predictive value chain. The results indicate that almost half the current engineering effort dedicated to manually checking the potential issues can now be automated, thus reducing the lead time to take a timely maintenance action, and ultimately optimizing the activities of the workforce.

The future work that is currently envisaged may further deal with the following topics:

- The explicit consideration of a “good” condition class to better understand the whole image degradation spectrum of the wheel tread defects. Although in the current scenario this is not strictly necessary because the maintenance staff already applies their criteria to take a picture, if this additional assessment was managed as a separate anomaly detection step prior to the described analysis, the whole pipeline would introduce a kind of double-check procedure.
- The collection of actual feedback from the field and the evaluation of the value added by the diagnosis. The Appendix shows some additional examples obtained with the minimum-viable product that is derived from the industrialization of the proposed solution. System interface feedback is also included in the continuous improvement of this online tool.
- The utility expansion to other types of wheels. Despite the proposed solution is tailored to steel railway wheelsets, the same technology can be applied to other types of wheels because CNNs ultimately tend to focus on their texture (Hermann, K. L., Chen, T., and Kornblith, S., 2020). For example, rubber-based tires would display patterns of deflation, punctures, tears or bulges on the

sidewalls, etc.

- In terms of safety, the proposed modular system is advantageous because the EN 50126-1 international railway standard specifies that such systemic hierarchy enables the assessment of subsystem interactions (CENELEC, 2017), and this is a prerequisite to understanding its overall limitations.
- In terms of security in a Deep Learning environment for Computer Vision, further robustness to adversarial images should also be studied, in addition to other cybersecurity considerations. In this sense, technologies like the Digital Twin enables virtual representations of components and systems (Moyné, J., Balta, E. C., Kovalenko, I., Faris, J., Barton, K., and Tilbury, D. M., 2020), which can help detect the presence of anomalous behaviors driven by attacks.

## ACKNOWLEDGMENT

We would like to show our gratitude to Fahd Janjua for his help with the data and engineering expertise, Joaquim Serra for his support with the low-level image processing tools, Verónica Fernández for her effort with the Self Organizing Map and the sensitivity analysis, Sergi Bermejo, Guillermo Sospedra and Vicente Fuerte for their management advice, and Alstom’s Innovation Board for funding this project. The contribution of Alexandre Trilla to this research was partially supported by the Government of Catalonia (Generalitat de Catalunya) Grant No. 2020 DI 54.

## APPENDIX

Additional examples of actual wheel tread defects along with their diagnostics are shown in Figure 12 and Figure 13.



Figure 12. Picture of a mild spall defect.

## REFERENCES

- Aji, I. P., and Kusuma, G. P. (2020). Landmark Classification Service Using Convolutional Neural Network and Ku-



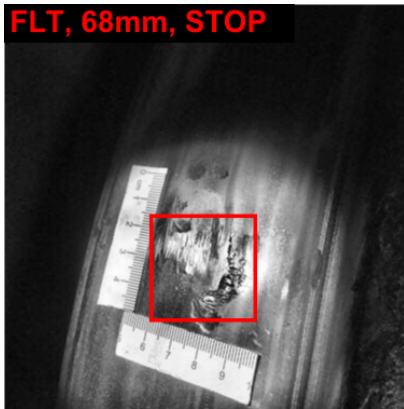


Figure 13. Picture of a critical flat defect.

bernetes. *International Journal of Advanced Trends in Computer Science and Engineering*, 9(3), 2817–2823.

- Alessi, A., La-Cascia, P., Lamoureux, B., Pugnaroni, M., and Dersin, P. (2016). Health Assessment of Railway Turnouts: A Case Study. *Proc. of the Third European Conference of the Prognostics and Health Management Society*, 1–8.
- Alsallakh, B., Jourabloo, A., Ye, M., Liu, X., and Ren, L. (2018). Do Convolutional Neural Networks Learn Class Hierarchy? *IEEE Transactions on Visualization and Computer Graphics*, 4, 1–11.
- Ammann, O., Michau, G., and Fink, O. (2020). Anomaly Detection And Classification In Time Series With Kernel Convolutional Neural Networks. *Proc. of the 30th European Safety and Reliability Conference and the 15th Probabilistic Safety Assessment and Management Conference*, 1–8.
- Bach, S., Binder, A., Montavon, G., Klauschen, F., Müller, K.-R., and Samek, W. (2015). On Pixel-Wise Explanations for Non-Linear Classifier Decisions by Layer-Wise Relevance Propagation. *PLoS ONE*, 10(7), 1–46.
- Balla, D., Maliosz, M., and Simon, C. (2020). Open Source FaaS Performance Aspects. *Proc. of the 43rd International Conference on Telecommunications and Signal Processing*, 358–364.
- Bau, D., Zhou, B., Khosla, A., Oliva, A., and Torralba, A. (2017). Network Dissection: Quantifying Interpretability of Deep Visual Representations. *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, 6541–6549.
- Bengio, Y. (2009). Learning Deep Architectures for AI. *Foundations and Trends in Machine Learning*, 2(1), 1–71.
- Biørn-Hansen, A., Majchrzak, T. A., and Grønli, T.-M. (2017). Progressive web apps: The possible web-native unifier for mobile development. *Proc. of the International Conference on Web Information Systems and Technologies*, 344–351.
- CENELEC. (2017). *Railway Applications - The Specification and Demonstration of Reliability, Availability, Maintainability and Safety (RAMS) - Part 1: Generic RAMS Process* (Tech. Rep. No. 50126-1:2017). European Committee for Electrotechnical Standardization.
- Chanthakit, S., Keeratiwintakorn, P., and Rattanapoka, C. (2019). An IoT System Design with Real-Time Stream Processing and Data Flow Integration. *Research, Invention, and Innovation Congress*, 1–5.
- Chen, Z., Bei, Y., and Rudin, C. (2020). Concept Whitening for Interpretable Image Recognition. *Nature Machine Intelligence*, 2, 772–782.
- Cui, Y., Kara, S., and Chan, K. C. (2020). Manufacturing big data ecosystem: A systematic literature review. *Robotics and computer-integrated Manufacturing*, 62(101861).
- Cybenko, G. (1989). Approximation by superpositions of a sigmoidal function. *Mathematics of Control, Signals and Systems*, 2, 303–314.
- Donahue, J., Jia, Y., Vinyals, O., Hoffman, J., Zhang, N., Tzeng, E., and Darrell, T. (2013). DeCAF: A Deep Convolutional Activation Feature for Generic Visual Recognition. *arXiv:1310.1531 [cs.CV]*, 1–10.
- Dubitzky, W., Granzow, M., and Berrar, D. (2007). Fundamentals of data mining in genomics and proteomics. *Springer Science & Business Media*, 178.
- Duda, R. O., Hart, P. E., and Stork, D. G. (Ed.). (2001). *Pattern Classification*. Wiley-Interscience.
- Eickenberg, M., Gramfort, A., Varoquaux, G., and Thirion, B. (2017). Seeing it all: Convolutional network layers map the function of the human visual system. *NeuroImage*, 152, 184–194.
- Engstrom, L., Tran, B., Tsipras, D., Schmidt, L., and Madry, A. (2019). Exploring the Landscape of Spatial Robustness. *Proc. of the 36th International Conference on Machine Learning*, 1–23.
- Erhan, D., Bengio, Y., Courville, A., and Vincent, P. (2009). Visualizing Higher-Layer Features of a Deep Network. *University of Montreal - Technical Report 1341*, 1–14.
- Fink, O., Wang, Q., Svensén, M., Dersin, P., Lee, W.-J., and Ducoffe, M. (2020). Potential, challenges and future directions for deep learning in prognostics and health management applications. *Engineering Applications of Artificial Intelligence*, 92(103678), 1–15.
- Fong, R., and Vedaldi, A. (2017). Interpretable Explanations of Black Boxes by Meaningful Perturbation. *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, 3429–3437.
- Globerson, T. C. H. S. A., A., & Roweis, S. (2009). An adversarial view of covariate shift and a minimax approach. In S. M. S. A. Quiñero-Candela J. & N. D. Lawrence (Eds.), *Dataset shift in machine learning* (pp. 179–197). Cambridge, Massachusetts: The MIT Press.
- Glorot, X., Bordes, A., and Bengio, Y. (2011). Deep sparse rectifier neural networks. *Proc. of the 14th International*

- Conference on Artificial Intelligence and Statistics*, 315–323.
- Goodfellow, I. J., Shlens, J., and Szegedy, C. (2015). Explaining and Harnessing Adversarial Examples. *Proc. of the International Conference on Learning Representations*, 1–11.
- Guo, G., Peng, J., Yang, K., Xie, L., and Song, W. (2017). Wheel Tread Defects Inspection Based on SVM. *Far East NDT New Technology Application Forum*, 251–253.
- Guo, Q., Chen, S., Xie, X., Ma, L., Hu, Q., Liu, H., Liu, Y., Zhao, J., and Li, X. (2019). An Empirical Study towards Characterizing Deep Learning Development and Deployment across Different Frameworks and Platforms. *Proc. of the IEEE/ACM International Conference on Automated Software Engineering*, 810–822.
- Han, K., Sun, M., Zhou, X., Zhang, G., Dang, H., and Liu, Z. (2017). A new method in wheel hub surface defect detection: Object detection algorithm based on deep learning. *Proc. of the International Conference on Advanced Mechatronic Systems*, 335–338.
- Hastings, E. M., Sexton, T., Brundage, M. P., and Hodkiewicz, M. (2019). Agreement Behavior of Isolated Annotators for Maintenance Work-Order Data Mining. *Proc. of the Annual Conference of the Prognostics and Health Management Society*, 1–7.
- He, K., Zhang, X., Ren, S., and Sun, J. (2015). Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37, 1904–1916.
- Hendrycks, D., and Dietterich, T. (2019). Benchmarking Neural Network Robustness to Common Corruptions and Perturbations. *Proc. of the International Conference on Learning Representations*, 1–16.
- Hermann, K. L., Chen, T., and Kornblith, S. (2020). The Origins and Prevalence of Texture Bias in Convolutional Neural Networks. *Proc. of the 34th Conference on Neural Information Processing Systems*, 1–25.
- Hinton, G. E., Srivastava, N., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. R. (2012). Improving neural networks by preventing co-adaptation of feature detectors. *arXiv:1207.0580 [cs.NE]*, 1–18.
- Hyde, P., Ulianov, C., and Defossez, F. (2016). Development and testing of an automatic remote condition monitoring system for train wheels. *IET Intelligent Transport Systems*, 10(1), 32–40.
- Ilyas, A., Santurkar, S., Tsipras, D., Engstrom, L., Tran, B., and Madry, A. (2019). Adversarial Examples Are Not Bugs, They Are Features. *Proc. of the 33rd Conference on Neural Information Processing Systems*, 1–12.
- ISO. (2003). *Condition monitoring and diagnostics of machine systems: Data processing, communication and presentation* (Tech. Rep. No. 13374-1:2003). International Organization for Standardization.
- Jang, R.-Y., Lee, R., Park, M.-W., and Lee, S.-H. (2020). Development of an AI Analysis Service System based on OpenFaaS. *The Journal of the Korea Contents Association*, 20(7), 97–106.
- Jo, J., and Bengio, Y. (2017). Measuring the tendency of CNNs to Learn Surface Statistical Regularities. *arXiv:1711.11561 [cs.LG]*, 1–13.
- Johnston, C. (2020). *Data Lakes*. Berkeley, CA, USA: Apress.
- Juba, S., and Volkov, A. (2019). *Learning PostgreSQL 11: a beginner's guide to building high-performance PostgreSQL database solutions*. Packt Publishing Ltd.
- Khan, A., Sohail, A., Zahoor, U., and Qureshi, A. S. (2020). A survey of the recent architectures of deep convolutional neural networks. *Artificial Intelligence Review*, 1–70.
- Kim, B., Kim, H., Kim, K., Kim, S., and Kim, J. (2019). Learning Not to Learn: Training Deep Neural Networks with Biased Data. *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, 9012–9020.
- Kohonen, T. (1990). The Self-Organizing Map. *Proc. of the IEEE*, 78(9), 1464–1480.
- Kornblith, S., Norouzi, M., Lee, H., and Hinton, G. (2019). Similarity of Neural Network Representations Revisited. *Proc. of the International Conference on Machine Learning*, 1–20.
- Kornblith, S., Shlens, J., and Le, Q. V. (2019). Do Better ImageNet Models Transfer Better? *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, 2661–2671.
- Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). ImageNet Classification with Deep Convolutional Neural Networks. *Advances in Neural Information Processing Systems*, 25, 1097–1105.
- Krummenacher, G., Ong, C. S., Koller, S., Kobayashi, S., and Buhmann, M. (2018). Wheel Defect Detection With Machine Learning. *IEEE Transactions on Intelligent Transportation Systems*, 19(4), 1176–1187.
- Le, H., and Borji, A. (2017). What are the Receptive, Effective Receptive, and Projective Fields of Neurons in Convolutional Neural Networks? *arXiv:1705.07049 [cs.CV]*, 1–7.
- LeCun, Y. and Bengio, Y., and Hinton, G. E. (2015). Deep Learning. *Nature*, 521, 436–444.
- Lecun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11), 2278–2324.
- Lee, K.-H., He, X., Zhang, L., and Yang, L. (2018). CleanNet: Transfer Learning for Scalable Image Classifier Training With Label Noise. *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, 5447–5456.
- Ma, K., Vicente, T. F. Y., Samaras, D., Petrucci, M., and Magnus, D. L. (2016). Texture classification for rail surface condition evaluation. *Proc. of the IEEE Winter Conference on Applications of Computer Vision*, 1–9.



- Magel, E., and Kalousek, J. (1996). Identifying and Interpreting Railway Wheel Defects. *Proc. of the International Heavy Haul Association Conference on Freight Car Trucks/Bogies*, 7–20.
- Mahendran, A., and Vedaldi, A. (2015). Understanding Deep Image Representations by Inverting Them. *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, 5188–5196.
- Masson, É., and Gransart, C. (2017). Cyber Security for Railways - A Huge Challenge - Shift2Rail Perspective. *Proc. of the International Workshop on Communication Technologies for Vehicles. Lecture Notes in Computer Science*, 10222, 97–104.
- Morcos, A. S., Barrett, D. G., Rabinowitz, N. C., and Botvinick, M. (2018). On the importance of single directions for generalization. *Proc. of the International Conference on Learning Representations*, 1–15.
- Moyne, J., Balta, E. C., Kovalenko, I., Faris, J., Barton, K., and Tilbury, D. M. (2020). A Requirements Driven Digital Twin Framework: Specification and Opportunities. *IEEE Access*, 8, 107781–107801.
- Nair, V., and Hinton, G. E. (2010). Rectified linear units improve restricted boltzmann machines. *Proc. of the 27th International Conference on Machine Learning*, 1–8.
- Nguyen, A., Yosinski, J., and Clune, J. (2016). Multifaceted Feature Visualization: Uncovering the Different Types of Features Learned by Each Neuron in Deep Neural Networks. *arXiv:1602.03616 [cs.NE]*, 1–23.
- Nikolenko, S. I. (2019). Synthetic Data for Deep Learning. *arXiv:1909.11512 [cs.LG]*, 1–156.
- Oquab, M., Bottou, L., Laptev, I., and Sivic, J. (2014). Learning and Transferring Mid-level Image Representations Using Convolutional Neural Networks. *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, 1717–1724.
- Pham, H., Dai, Z., Xie, Q., Luong, M.-T., and Le, Q. V. (2020). Meta Pseudo Labels. *arXiv:2003.10580 [cs.LG]*, 1–22.
- Pinkus, A. (1999). Approximation theory of the MLP model in neural networks. *Acta Numerica*, 8, 143–195.
- Pizer, S. M., Amburn, E. P., Austin, J. D., Cromartie, R., Geselowitz, A., Greer, T., Romeny, B. t. H., Zimmerman, J. B., and Zuiderveld, K. (1987). Adaptive Histogram Equalization and Its Variations. *Computer Vision, Graphics, and Image Processing*, 39, 355–368.
- Razavian, A. S., Azizpour, H., Sullivan, J., and Carlsson, S. (2014). CNN Features Off-the-Shelf: An Astounding Baseline for Recognition. *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, 512–519.
- Real, E., Moore, S., Selle, A., Saxena, S., Suematsu, Y. L., Tan, J., Le, Q. V., and Kurakin, A. (2017). Large-Scale Evolution of Image Classifiers. *Proc. of the 34th International Conference on Machine Learning*, 70, 2902–2911.
- Rezaeianjouybari, B., & Shang, Y. (2020). Deep learning for prognostics and health management: State of the art, challenges, and opportunities. *Measurement*, 163(107929).
- Ricci, M., Kim, J., and Serre, T. (2018). Same-different problems strain convolutional neural networks. *arXiv:1802.03390 [cs.CV]*, 1–6.
- Salman, H., Ilyas, A., Engstrom, L., Kapoor, A., and Madry, A. (2020). Do Adversarially Robust ImageNet Models Transfer Better? *Proc. of the 34th Conference on Neural Information Processing Systems*, 1–31.
- Sermanet, P., Eigen, D., Zhang, X., Mathieu, M., Fergus, R., and LeCun, Y. (2013). OverFeat: Integrated Recognition, Localization and Detection using Convolutional Networks. *arXiv:1312.6229 [cs.CV]*, 1–16.
- Settles, B. (2010). Active Learning Literature Survey. *University of Wisconsin–Madison - Computer Sciences Technical Report 1648*, 1–67.
- Shang, L., Yang, Q., Wang, J., Li, S., and Lei, W. (2018). Detection of rail surface defects based on CNN image recognition and classification. *Proc. of the International Conference on Advanced Communication Technology*, 45–51.
- Shojaefard, M. H., Akbari, M. Tahani, M., and Farhani, F. (2013). Sensitivity Analysis of the Artificial Neural Network Outputs in Friction Stir Lap Joining of Aluminum to Brass. *Advances in Materials Science and Engineering*, 1–7.
- Simard, P. Y., Steinkraus, D., and Platt, J. C. (2003). Best Practices for Convolutional Neural Networks Applied to Visual Document Analysis. *Proc. of the Seventh International Conference on Document Analysis and Recognition*, 1–6.
- Simonyan, K., and Zisserman, A. (2015). Very Deep Convolutional Networks for Large-Scale Image Recognition. *Proc. of the International Conference on Learning Representations*, 1–14.
- Simonyan, K., Vedaldi, A., and Zisserman, A. (2013). Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps. *arXiv:1312.6034 [cv.CV]*, 1–8.
- Springenberg, J. T., Dosovitskiy, A., Brox, T., and Riedmiller, M. (2015). Striving for Simplicity: The All Convolutional Net. *Proc. of the International Conference on Learning Representations*, 1–14.
- Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. (2014). Dropout: a simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15, 1929–1958.
- Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., and Rabinovich, A. (2015). Going deeper with convolutions. *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, 1–9.
- Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., and Wojna,

- Z. (2016). Rethinking the Inception Architecture for Computer Vision. *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, 2818–2826.
- Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., and Fergus, R. (2013). Intriguing properties of neural networks. *arXiv:1312.6199 [cs.CV]*, 1–10.
- Tan, M., and Quoc, V. L. (2019). EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks. *Proc. of the International Conference on Machine Learning*, 1–10.
- Tian, J., Azarian, M. H., and Pecht, M. (2014). Anomaly Detection Using Self-Organizing Maps-Based K-Nearest Neighbor Algorithm. *Proc. of the European Conference of the Prognostics and Health Management Society*, 1–9.
- Vickerstaff, A., Bevan, A., and Boyacioglu, P. (2020). Predictive Wheel-rail Management in London Underground: Validation and Verification. *Proc. of the Institution of Mechanical Engineers, Part F: Journal of Rail and Rapid Transit*, 234(4), 393–404.
- Wei, D., Zhou, B., Torralba, A., and Freeman, W. T. (2015). Understanding Intra-Class Knowledge Inside CNN. *arXiv:1507.02379 [cs.CV]*, 1–7.
- Wu, C., Haihong, E., and Song, M. (2020). An Automatic Artificial Intelligence Training Platform Based on Kubernetes. *Proc. of the 2nd International Conference on Big Data Engineering and Technology*, 58–62.
- Yang, Y., and Xu, Z. (2020). Rethinking the Value of Labels for Improving Class-Imbalanced Learning. *Proc. of the 34th Conference on Neural Information Processing Systems*, 1–21.
- Yosinski, J., Clune, J., Bengio, Y., and Lipson, H. (2014). How Transferable Are Features in Deep Neural Networks? *Proc. of the 27th International Conference on Neural Information Processing Systems*, 2, 3320–3328.
- Yusuf, S. (2016). *Ionic Framework By Example*. Packt Publishing Ltd.
- Zeiler, M. D., and Fergus, R. (2014). Visualizing and Understanding Convolutional Networks. *Proc. of the European Conference on Computer Vision. Lecture Notes in Computer Science*, 8689, 818–833.
- Zhang, J., Guo, Z., Jiao, T., and Wang, M. (2018). Defect Detection of Aluminum Alloy Wheels in Radiography Images Using Adaptive Threshold and Morphological Reconstruction. *Applied Sciences*, 8(2365), 1–12.
- Zhang, L., Song, J., Gao, A., Chen, J., Bao, J., and Ma, K. (2019). Be Your Own Teacher: Improve the Performance of Convolutional Neural Networks via Self Distillation. *arXiv:1905.08094 [cs.LG]*, 1–10.
- Zhang, W., Zhang, Y., Li, J., Gao, X., and Wang, L. (2014). The defects recognition of wheel tread based on linear CCD. *IEEE Far East Forum on Nondestructive Evaluation/Testing*, 302–307.
- Zhang, Y., Cui, X., Liu, Y., and Yu, B. (2018). Tire Defects Classification Using Convolution Architecture for Fast Feature Embedding. *International Journal of Computational Intelligence Systems*, 11, 1056–1066.
- Zhao, W., Siegel, D., Lee, J., and Su, L. (2013). An Integrated Framework of Drivetrain Degradation Assessment and Fault Localization for Offshore Wind Turbines. *International Journal of Prognostics and Health Management*, 1–13.
- Zhou, B., Khosla, A., Lapedriza, A., Oliva, A., and Torralba, A. (2015). Object detectors emerge in Deep Scene CNNs. *Proc. of the International Conference on Learning Representations*, 1–12.

## BIOGRAPHIES

**Alexandre Trilla** graduated from La Salle University of Barcelona with a M.Sc. in Electronics and Telecommunications Engineering in 2008, and a M.Sc. in IT Management in 2010. He has an academic research background in spoken language processing, and an industrial research background in PHM. He has authored several publications in scientific conferences and journals (IEEE Transactions on Audio, Speech, and Language Processing, Chemical Engineering Transactions, and the Journal of Rail and Rapid Transit). At present, he is a Senior Data Scientist and R&D Program Manager at Alstom, working on the deployment of PHM to the railway environment. He leads the development of predictive maintenance based on Machine Learning, and he is especially interested in building solutions with artificial neural networks.

**John Bob-Manuel** is a Mechanical Systems Engineer at Alstom, where he is responsible for the performance of the mechanical systems of London Underground's Northern Line fleet in order to maintain and improve the reliability, availability, and safety of the trains. John obtained his M.Sc. in Advanced Mechanical Engineering from Imperial College London, and received his Bachelor's degree in Aerospace Engineering from Queen Mary University of London. Before joining Alstom, he worked in the oil & gas and aerospace industries, where he led multi-disciplinary engineering teams to design and analyze systems ranging from remote, deep offshore structures to aircraft interior systems. He has a keen interest in using data science to help inform business decisions in specific areas of engineering operations.

**Benjamin Lamoureux** is a Data Scientist and Predictive Maintenance engineer at Alstom. Dr. Lamoureux has received both a Master's Degree in Mechatronics from the University Pierre & Marie Curie and a Master's Degree in Engineering from Arts & Métiers ParisTech in Paris in 2010. From 2011 to 2014, he performed an industrial Ph.D. work in collaboration between Arts & Métiers ParisTech and SAFRAN Aircraft Engines Villaroche (formerly SAFRAN Snecma). He received his Ph.D. in June 2014. In October 2014, he joined the PHM department of Alstom Saint-Ouen to extend his Ph.D. research

in the railway domain, and he still occupies the same position today. His current research interests are machine learning, statistics, data science and computer science applied to PHM.

**Xavier Vilasis-Cardona** is full professor at La Salle, Universitat Ramon Llull, Barcelona. He holds a degree in physics

(’89) and a PhD in physics (’93) by Universitat de Barcelona. He is member of the IEEE, of the IEEE CNNAC technical committee and of the LHCb collaboration. He is currently leading the Data Science for the Digital Society (DS4DS) research group.