RESEARCH ARTICLE

Automatic detection and classification of manufacturing defects in metal boxes using deep neural networks

Oumayma Essid[1], **Hamid Laga**[2,3], **Chafik Samir**[1]*

1 CNRS LIMOS UMR 6158, University of Clermont Auvergne, France, **2** School of Engineering and Information Technology, Murdoch University, Perth, Australia, **3** Phenomics and Bioinformatics Research Centre, University of South Australia, Australia

* chafik.samir@uca.fr

Citation: Essid O, Laga H, Samir C (2018) Automatic detection and classification of manufacturing defects in metal boxes using deep neural networks. PLoS ONE 13(11): e0203192. https://doi.org/10.1371/journal.pone.0203192

Editor: Zhaoqing Pan, Nanjing University of Information Science and Technology, CHINA

Received: October 24, 2017

Accepted: July 27, 2018

Published: November 9, 2018

Data Availability Statement: All data underlying the study are within the paper and its Supporting Information files.

Funding: This work was supported by University of Clermont Auvergne, University of Murdoch, (CS). The funders had no role in study design, data collection and analysis, decision to publish, or preparation of the manuscript.

Competing interests: The authors have declared that no competing interests exist.

Abstract

This paper develops a new machine vision framework for efficient detection and classification of manufacturing defects in metal boxes. Previous techniques, which are based on either visual inspection or on hand-crafted features, are both inaccurate and time consuming. In this paper, we show that by using autoencoder deep neural network (DNN) architecture, we are able to not only classify manufacturing defects, but also localize them with high accuracy. Compared to traditional techniques, DNNs are able to learn, in a supervised manner, the visual features that achieve the best performance. Our experiments on a database of real images demonstrate that our approach overcomes the state-of-the-art while remaining computationally competitive.

1 Introduction

Automatic inspection and defect detection using image processing is an area of machine vision that is being widely adopted in many industries. It is used for high throughput quality control in production systems such as the detection of flaws on manufactured surfaces, e.g. metallic rails [1] or steel surfaces [2]. The idea is to design autonomous devices that automatically detect and examine specific visual patterns from images and videos in order to overcome the limitations and improve the performance of the traditional inspection systems that depend heavily on human inspectors.

Various systems have been previously developed for the automatic inspection of surfaces of different products. They are generally composed of a pipeline of several steps, each one introduces a set of challenges. First, the acquisition step requires efficient calibration of various types of sensors such as cameras and lighting systems [3, 4]. The quality of the images produced by the acquisition systems impacts directly the performance of the subsequent analysis steps.

Once images are acquired, the second step is the extraction of visual features, which are then used as input to classifiers, which return the likelihood of the presence of a defect at each pixel of the image. These classifiers can be supervised or non-supervised. The main challenge here is two-fold; first one has to design features that are suitable for the problem at hand. The second one is the design of classifiers that are able to learn the boundaries in the feature space that separate defects from non-defects. While both problems have been extensively investigated [5, 6], existing techniques still fail to achieve high performance especially under challenging conditions such as those present in industrial environments.

This paper is concerned with the design of deep learning techniques that are able to automatically detect and classify manufacturing defects in the surface of metal boxes. The advantage of using deep learning techniques is their ability to learn, in a supervised manner, the visual features that achieve the best detection and classification performance. We show in the experimental part that the proposed framework outperforms traditional techniques that are based on hand-crafted features.

Real time defect detection, using image processing, has been extensively studied in the literature [7, 8]. Previous techniques can be divided into three general categories based on (1) their input (single vs. multiple images, RGB vs. depth images), (2) the type of features they extract from the input images, and (3) the type of classifiers they use for detecting defect regions and for classifying various types of defects. Ideally, one would like to detect and classify defects only from one single RGB image. However, this is not often an easy task especially when dealing with complex defects [9].

Once images are acquired, the next step is to extract visual features from them which will serve as input to some classifiers. Shen et al. [10], for example, segment images into regions, by grouping pixels of same intensity, and use some thresholding techniques to extract visual features. Shanmugamani et al. [9] used multiple textural features based on histogram and gray level co-occurrence combined with several classifiers based on Bayes, K-Nearest Neighbors (KNN), Artificial Neural Network (ANN), and Support Vector Machines (SVM) for detecting and classifying defects.

Once the features have been extracted, the last step is to learn a decision function, which returns the likelihood that a given feature corresponds to a defect region of a certain type. Such decision function is usually the output of a supervised or unsupervised classifier. Although several classifiers have been used, many previous works suggest that Support Vector Machines (SVM) [9] achieve better performance compared to Bayesian classifiers or Artificial Neural Networks (ANN) [5], especially when the classes that represent the various defects are separable.

While this widely used pipeline achieves satisfactory results, the main challenge in manufacturing systems is to achieve higher accuracies since improving the classification accuracy even with small percentages, can result in substantial economic gains. In fact, major advances can be achieved by letting the algorithms learn the discriminative features that optimally represent the data. This concept lies at the basis of many deep learning algorithms, which are models (networks) composed of many layers that transform input data (e.g. images) to outputs (e.g. binary decision true/false) while learning increasingly higher level features.

Convolutional Neural Networks (CNN) are one example of successful types of deep learning models. CNNs are composed of many layers. Each layer transforms its input by applying small convolution filters whose weights are learned in a supervised manner using large training samples. Despite their reliance on large training sets, deep learning has recently started delivering interesting results and showing impact in many applications [11, 12]. This is mainly due to the availability of large training datasets and powerful hardware that can accelerate the training process. The choice of a neural architecture depends on the application at hand. The most popular algorithms of deep learning are autoencoders [5, 11], deep belief networks, and convolutional neural networks [5, 13].

Since the first use of classical CNNs in inspection task [8, 14], they have shown a good classification accuracy for many applications such as the detection of defects on photometric stereo images of rail surfaces [1]. Another application of CNNs is the classification of steel images and used Pyramid of Histograms of Orientation Gradients (HOG) as feature extractor and Max-Pooling Convolutional Neural Networks. The approach achieves a high accuracy with 7% error rate, which is significantly better than other classifier such SVM [2].

Pretrained deep Convolutional Neural Networks have also achieved an excellent performance in image classification and recognition tasks. For instance, Kasthurirangan et al. [15] used pretrained CNN architectures to classify defects in crack pavement. First, some preprocessing methods were applied on the image dataset to enhance the information in the image. The pretrained VGG16 was used to extract, from images, features that can distinguish one image class from another. Different classifiers such as single-layer Neural Network (NN), SVM and Random Forest (RF) were tested proving the performance of (CNN) in this inspection task.

Autoencoder architectures have proved their efficiency with multi-modal data in many classification or retrieval tasks [11]. Their successors, stacked autoencoders [16], where each autoencoder is trained with the output of the hidden layer of the previous autoencoder in the stack. With this architecture, the level of complexity and abstraction of learned representation increase along the stack of autoencoders. Autoencoder trees, in analogy to decision trees, are a new form of neural networks that learn hierarchical representations at different resolutions where layers are replaced by tree's nodes [17, 18].

In this paper, we propose a framework for detecting defects on metal boxes and for classifying images into defective or non-defective. The proposed framework uses Deep Neural Networks, which can learn, in a supervised manner, the appropriate features (and thus image representation) as well as the classification function. First, we apply some preprocessing techniques on images of a metallic box to improve their quality. Then, we use an adapted method based on a multi-layer network for learning. Recall that using an autoencoder leads to dimensionality reduction since the number of neurons in hidden layers is smaller than that of input layer. The classification decision is made using a probabilistic model that takes the output of the last layer and returns the likelihood of the images being of a defected metallic box. Finally, we also propose an algorithm that detects the defect if present in the image.

2 Material and methods

1 Data acquisition

Metallic boxes or steel can are extensively exploited as containers for the distribution or storage of food. The metallic boxes used in this study were collected by a company who produces steel cans with or without colored printing. Collected metallic boxes are suitable for packing tuna, salmon, fish, shrimp, mushroom, and other foodstuffs. All images were acquired from an installed and configured image acquisition device using Raspberry Pi camera module and processed by OpenCV in unconstrained environment on the company's premises. For each metallic box, the device captures and stores two images that cover the entire box, see Fig 1. Such images can correspond to either non-defective (Fig 1(a)) or defective boxes. The defects can be small (Fig 1(b) and 1(c)) or big (Fig 1(d), 1(e) and 1(f)). In this paper, we consider that we have two different classes of images: defective and non-defective images.

2 Data description and representation

Due to non-controlled external factors, each step in the acquisition process may introduce noise (random changes) to the raw data (e.g. pixel values). To reduce the effects of such noise

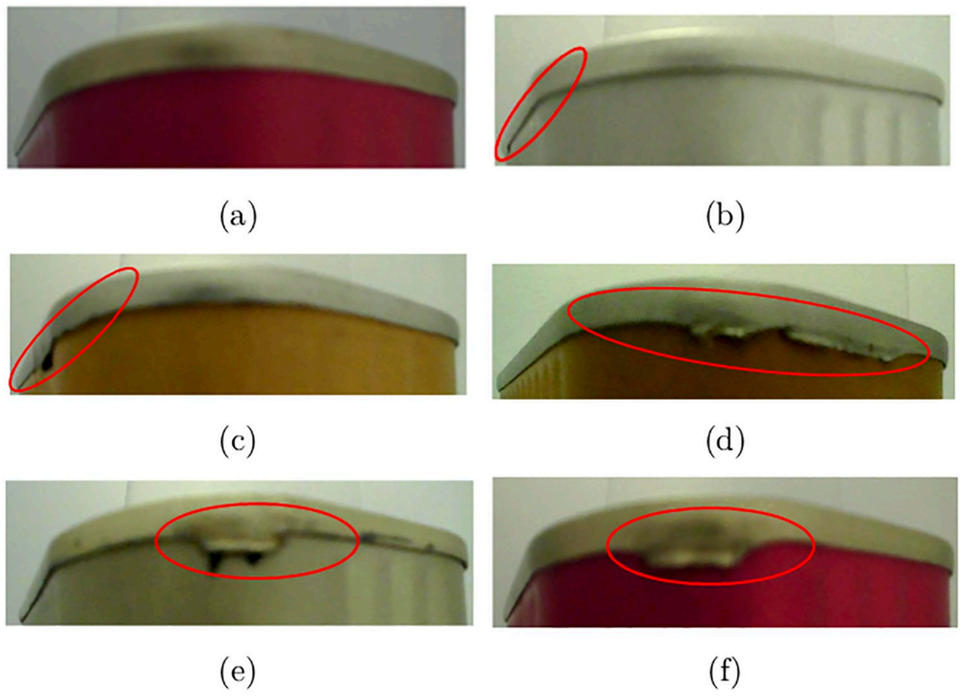


Fig 1. Examples of images: (a) without defect, (b)& (c) with small defects, and (d:f) with big defects.

https://doi.org/10.1371/journal.pone.0203192.g001

on the quality of the inspection process, we first pre-process the images by applying a set of denoising filters and reduce the spread by treating the database as a normal distribution of intensities [19, 20]. We refer to this step as *the image normalization process*. Another motivation for image normalization is to standardize the input of the autoencoder, used in the learning process, in order to reduce variability between intensity distributions [21].

Next, we observe that most of the defects are localized in areas that are dominated by horizontal edges, see Fig 1 for some examples. To highlight these regions, we use the gradient of

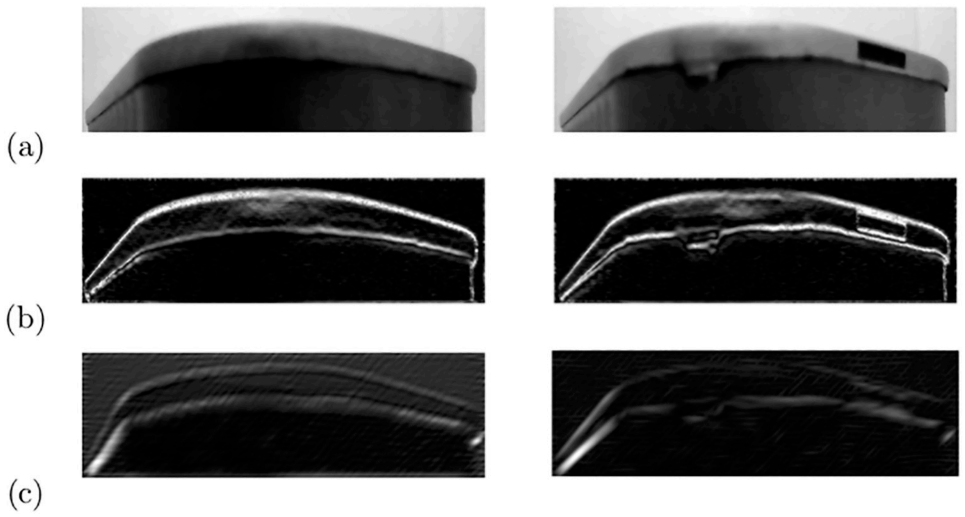


Fig 2. Examples of features extracted from two different images: (a) original images, (b) gradient, and (c) Gabor features.

https://doi.org/10.1371/journal.pone.0203192.g002

the images instead of the images themselves [22]. Other representations and features have been proposed in the literature and we will use some of them for evaluation in our experiments. Since we are interested in horizontal contours where most of the defects are localized, we extract features by applying a Gabor filter with a $\frac{\pi}{2}$ orientation. Fig 2 shows two examples of extracted features where the first row shows the input images, the second and the third rows show the gradient and the Gabor features, respectively. Another way to capture relevant features is to decompose the image using adaptive basis such as Fourier, wavelets, or polynomials. Wavelet decompositions have been successfully used for many tasks such as dimensionality reduction, image/video classification, and many other areas of image analysis [23]. This motivated their use for evaluation in our experiments which has shown to be successful.

3 Features learning using an autoencoder

An autoencoder neural network architecture is a feedforward network composed of one or multiple connected hidden layers. It uses a non linear mapping function between the original data as input and outputs specific learned features [24, 25]. Several previous works have used autoencoders for feature learning or for dimensionality reduction since the number of neurons in the hidden layer is smaller than the dimension of the input. In fact, many studies [26, 27] showed that an autoencoder with nonlinear characteristic is more efficient than linear dimensionality reduction techniques such as Principal Component Analysis (PCA) when the input data have a nonlinear or a sparse structure. Overall, this provides significant benefits for the inspection task: the computational time and the required memory for storage. And even better, removing the redundant information by maximizing the covariance.

Stacked autoencoders [16, 28] are deep neural architectures composed of a succession of autoencoders. Each autoencoder is trained with the output of the hidden layer of the previous autoencoder in the stack. This architecture, which can be seen as a convolution, allows learning complex concepts in an progressive manner. Consequently, the output representations are more relevant when one autoencoder is not sufficient to capture interesting structures that maximize the covariance between the components.

In this work, we will follow the same strategy; We train an autoencoder with different layers to reduce dimensionality as well as extracting relevant features under constraints formulated by a cost function. In the training phase, we used a sparse autoencoder whose training criterion involves a sparsity penalty. Given a finite set of N images, represented by vectors x_i with $i = 1...N$, and their corresponding labels $y_i \in \{0, 1\}$, the learning is done by stacking multiple layers and the cost function E is defined by:

$$E(X, Y, W, b) = \frac{1-\lambda}{2}\|Y - r(X, W, b)\|_2^2 + \frac{\lambda}{2}\|W\|_2^2, \qquad (1)$$

where r is the activation function, X and Y represent the observations, W and b are the network parameters, and $\|W\|_2^2$ is a regularization term. The Lagrangian parameter λ weights between bounds of W and proximity to the input.

This cost function is a deterministic functional on parametric functions of the form $r(X) = W^T X + b$ that can be minimized using numerical methods to search for optimal parameters $\hat{W}$ and $\hat{b}$. Focusing on neural networks, a wide variety of models can be learned by varying different factors such as the activation function, the number of hidden units, and the choice of the regularization term. The details of the parameters initialization and the optimization process have been discussed and studied in the literature without determining the best choices. We consider that the answer is out of the scope of this paper and we give details of our model in Section 3.

One important aspect about autoencoders is dimensionality reduction and the features computed at the hidden layers, which are useful for inspection tasks. If the output of the autoencoder is considered as a deterministic function, one can plug it into a probabilistic model and perform learning and regression within the same framework. This idea is the main motivation of the proposed method and we will show that it helps improving the performance of the classification.

The idea of learning relevant features from data automatically as an output of an autoencoder can be illustrated by visualizing the hidden layers. For an autoencoder with several layers, every layer encodes different features. For example, Fig 3 shows (a) the output of the second hidden layer of the autoencoder trained on a set of grayscale images, (b) their corresponding gradients, and (c) gradient after binarization. Note that at this stage, the classifier is not yet defined but we expect the output to highlight relevant features. Even if it is hard to judge which one is better, we can say that the area around contours is further enhanced for different inputs.

4 Regression using Gaussian processes

We assume that we have N observations (X_1, Y_1),..,(X_N, Y_N) of the same law of variables (X, Y). Here X is the observation (e.g. input image) and Y is its label. To be more specific, we restrict our analysis to the cases where X has a compact support in a finite-dimensional Euclidean space and Y is the class label with $y = 0$ for non-defective boxes and $y = 1$ for defective boxes.

Any regression model is based on building a predictive model: learn a probabilistic model from observed Y at different locations of X that will be able to predict y^*, the class label, of a new vector x^*. Before giving details of the proposed model, we first review classical regression methods, which consider X as elements on a high dimensional space. Then we provide a brief introduction to dimensionality reduction techniques. Next, we introduce the automatic learned features from an autoencoder as a parametric transformation of the input. Finally, we give details of Manifold Regression, the proposed method for jointly learning a regression model and a suitable feature representation ϕ of the data. Though mildly technical, it is useful as we focus on a particular subset of the relevant background in forming our model developments. We begin by reviewing a logistic regression model that has been initially introduced in [29]. The goal is to find the best fitting model representing the relationship between the output variable Y and a set of input variable $X = (X_1,.., X_p)$ where $Y \in \{0, 1\}$.

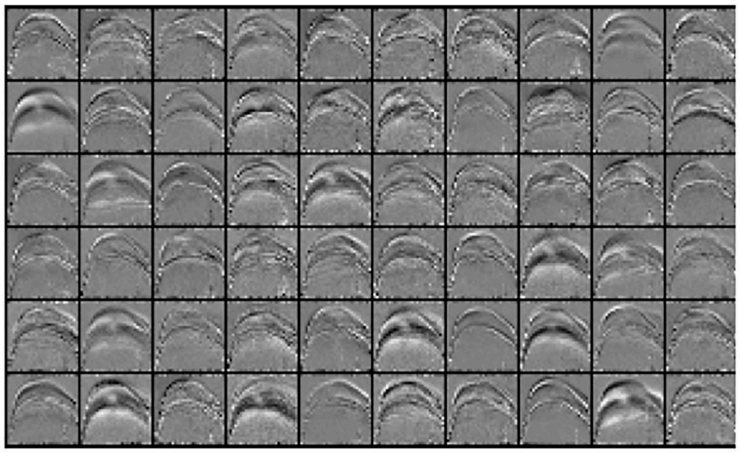
4.1 MLE-based regression. We suppose that we have N mutually independent samples (X_1, Y_1),..,(X_N, Y_N) of the same law as (X, Y). We consider the problem of fitting a logistic model $y = \sigma(x^T \beta)$ where σ is the sigmoid function which is equivalent to estimating β from observed samples. Let (x_i, y_i) be the observed value of the explanatory variables (X_i, Y_i) for each observation $i \in \{1,.., N\}$. We denote by $\pi_\beta(x_i)$ the probability of $Y_i = 1$ for a given $X = x_i$:

$$P(Y_i = 1|X = x_i) = \pi_\beta(x_i) = \frac{\exp(x_i{}^T\beta)}{1 + \exp(x_i{}^T\beta)} \tag{2}$$

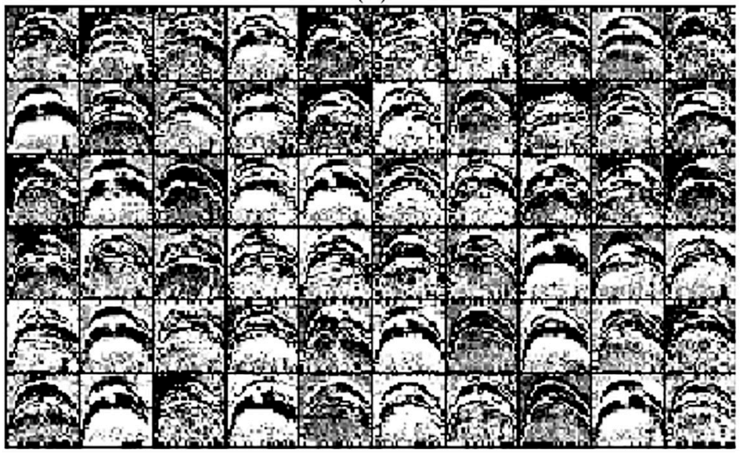
and by modeling $Y_i|X = x_i$ as a Bernoulli distribution $\pi_\beta(x_i)$, $Y_i|X = x_i \sim \mathcal{B}(\pi_\beta(x_i))$ we write the negative log-likelihood as:

$$l(\beta) = \sum_{i=1}^{N} y_i \log(\pi_\beta(x_i)) + (1 - y_i) \log(1 - \pi_\beta(x_i)) \tag{3}$$

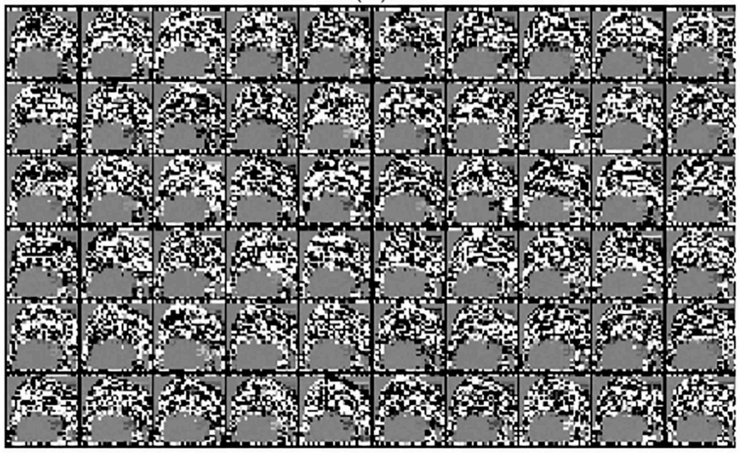
$$= \sum_{i=1}^{N} y_i x_i{}^T\beta - \log(1 + \exp(x_i{}^T\beta)) \tag{4}$$



(a)



(b)



(c)

Fig 3. Visualization of the output features from the second hidden layer when applying an autoencoder on: (a) grayscale images, (b) gradient images, and (c) the binary images.

https://doi.org/10.1371/journal.pone.0203192.g003

Then the gradient of l at β is given by:

$$\nabla l(\beta) = \sum_{i=1}^{N} x_i (y_i - \pi_\beta(x_i)) \tag{5}$$

$$= X^T (Y - \pi_\beta) \tag{6}$$

where $\pi_\beta = (\pi_\beta(x_1),.., \pi_\beta(x_N))^T$, $X = (x_1,.., x_N)^T$ and $Y = (y_1,.., y_N)^T$. Typically, to find the MLE we have to search for the critical point $\hat{\beta}$ of the gradient: $\nabla l(\hat{\beta}) = 0$. Then the maximization of $l(\beta)$ yields to the ordinary MLE β^*. If X is of maximum rank, we have $\beta \mapsto l(\beta)$ is strictly concave, i.e β^* exists and is unique [29]. We note that finding β^* explicitly is not straightforward, and consequently is very common to use iterative algorithms based on Newton and gradient descent methods.

4.2 Penalized MLE-based regression. Basically inspired from the Ridge logistic [29], the weighted logistic method is more suitable when the components of X are highly correlated or when we have a large number of explanatory variables. Remind that in both cases X is not of maximum rank which means that there is no guarantee for β^* to exist nor to be unique. In an effort to address this issue, the idea of ridge logistic consists in adding a regularization term. The regularization will have the effect of controlling the model and improving the performance in the presence of an over-fitting. Consequently, we consider a modified version of Eq 5.

$$l^\lambda(\beta) = \frac{(1-\lambda)}{2} l(\beta) - \frac{\lambda}{2} ||\beta||_2^2 \tag{7}$$

where the regularization parameter satisfies: $0 < \lambda < 1$. We denote by $\beta^{\lambda,*}$ the optimal solution. Therefore for a better choice of λ, the estimator $\beta^{\lambda,*}$ should maximize the log-likelihood compared to the unstructured MLE, ie. $\mathrm{MSE}(\beta^{\lambda,*}) < \mathrm{MSE}(\beta^*)$. Following the same steps for the new modified cost, the gradient of $l^\lambda(\beta)$ is:

$$\nabla l^\lambda(\beta) = \frac{(1-\lambda)}{2} \nabla l(\beta) - \lambda\beta \tag{8}$$

with $\nabla l(\beta)$ is the gradient of $l(\beta)$ as detailed in Eq 5. Then the estimator is a solution of $\nabla l^\lambda(\beta) = 0$ and the negative Hessian of $l^\lambda(\beta)$ is

$$H^\lambda(\beta) = \frac{(1-\lambda)}{2} H(\beta) + \lambda I \tag{9}$$

with

$$H(\beta) = -\sum_{i=1}^{N} x_i x_i^T \pi_\beta(x_i)(1 - \pi_\beta(x_i)) \tag{10}$$

As for the previous optimization formulation we use an iterative approach based on Newton method. For numerical efficiency, we present an approximation of the Newton-MLE based on Taylor expansion of $\nabla l^\lambda(\beta^1)$ at β^0:

$$\nabla l^\lambda(\beta^1) \approx \nabla l^\lambda(\beta^0) - (\beta^1 - \beta^0)^T H^\lambda(\beta^0) \tag{11}$$

In particular, we have $\nabla l^\lambda(\beta^1) = 0$ and a first-order approximation for β^1 as:

$$\beta^1 \approx \beta^0 + (\frac{(1-\lambda)}{2} H^\lambda(\beta^0) + \lambda I)^{-1} (\frac{(1-\lambda)}{2} \nabla l^\lambda(\beta^0) - \lambda\beta) \tag{12}$$

This process is then updated iteratively until convergence.

$$\beta^{k+1} \approx \frac{(1-\lambda)}{2}(\frac{(1-\lambda)}{2}H(\beta^k) + \lambda I)^{-1}(H(\beta^k) + \nabla l(\beta^k)) \tag{13}$$

5 Gaussian processes classifier

In this section we make connection with binary classification, as described in the previous section, which forms the foundation of Gaussian processes (GP). Gaussian Processes are a state-of-the-art non-parametric probabilistic regression method. In order to capture the correlation between observed *X*s, build a probabilistic model and perform optimal predictions for non-observed data, we study a GP as a distribution over the transformed variables $\phi \sim GP(m, C)$ and fully defined by a mean function m (in our case $m = 0$) and a covariance function C. The popularity of such processes stems primarily from two essential properties. First, a Gaussian process is completely determined by its mean and covariance functions. This property facilitates model fitting as only the first- and second-order moments of the process require specification. Second, solving the prediction problem is straightforward since the optimal predictor at an unobserved position is a linear function of the observed values.

In the simple case of a real random process, $Y(x), x \in \mathbb{R}$ is a Gaussian process if all the finite-dimensional distributions have a multivariate normal distribution. That is, for distinct observations $x_1, x_2, \ldots, x_n$, the random vector $Y_1, Y_2, \ldots, Y_n$ with $Y_i = Y(x_i)$ has a multivariate normal distribution with mean vector $m = E[Y]$ and covariance matrix C with $C_{i,j} = C(Y_i, Y_j)$. A Gaussian process is said to be stationary if m is independent of x and the covariance $C(Y(x+h), Y(x)) = C(h) < \infty$ is independent of x (i.e., the process $Y(x)$ is translation invariant). This condition is usually called the strong form of stationarity (second-order or weak). Considering the new formulation, we introduce a new latent variable ϕ with:

$$\pi = \mathbb{P}(Y|X = x) = \sigma(\phi(x)) \tag{14}$$

The idea behind GP prediction is based on placing a GP prior on ϕ, i.e: $\phi \sim \mathcal{GP}(m, C_\theta)$. Where m is the mean, equal to zero in our case, and C_θ is a family of covariance functions with parameter θ. There is a wide choice of covariance functions but, as in this study, the Matérn covariance functions is a preferred choice due its smoothness and asymptotic properties [30, 31]. If we consider prediction from the observed locations. Obviously, we are looking for a prediction method that works well on the average. One of the main difficulties is the choice of the covariance function. Despite the fact that a prediction model may yield to an unbiased predictor with a correct predictive variance, it is only true if the choice of a covariance function is optimal. To deal with such issue, a common choice consists in parametric covariance functions leading to an search for an optimal parameter $\hat{\theta}$. Many numerical methods have been used to search for $\hat{\theta}$ and the most studied and popular one is Maximum Likelihood Estimator (MLE).

We keep the same notations from the previous section and by abuse of notation, we note $\phi = (\phi_1,.., \phi_N)^T = (\phi(x_1),.., \phi(x_N))^T$. Therefore, the goal is to estimate the hyperparameter $\theta = (\alpha, \tau, \nu)$ of the covariance function C_θ that minimizes the negative likelihood L_θ:

$$L_N(\theta|\phi) \quad = \quad \frac{N}{2\pi}\ln(2\pi) + \frac{N}{2}\ln(\tau) + \frac{1}{2\tau}\ln(\det C_\theta) + \frac{1}{2}\phi^T C_\theta^{-1}\phi \tag{15}$$

where $C_{\theta,i,j} = C_\theta(d(\phi_i, \phi_j))$ is the covariance matrix for $\phi_1, \phi_2, \ldots, \phi_n$ and θ is the full parameter vector taking values in the parameter space $\Theta = \{\tau > 0, \alpha > 0, \nu = 1 + k, k \in \mathbb{N}\}$. Our goal is then to find the maximum likelihood estimator (MLE) $\hat{\theta}$ of θ. Once, there is no analytical solution for Eq 15, we use a Newton-based method to find the MLE. The optimization problem over v is not straightforward in this case, and thus, we estimate this value using a k-fold cross-validation rather than the MLE.

To summarize, the GP predictive distribution at a new observation $\phi^* = \phi(X^*)$ is given by

$$\hat{\mathbb{P}}(\phi^*|X, Y, X^*) = \mathcal{N}(\mu(X^*), \sigma^2(X^*)) \tag{16}$$

$$\mu(X^*) = c_*^T C^{-1}\hat{\phi} \tag{17}$$

$$\sigma^2(X^*) = c_{**} - c_*^T(C + W^{-1})^{-1}c_* \tag{18}$$

where $c_{**} = C(X^*, X^*)$ and $c_* = C(X, X^*)$ and W is a $N \times N$ diagonal matrix with $W_{ii} = \frac{\exp(\phi_i)}{1+\exp(\phi_i)}$. Given the mean and the variance, we make predictions by computing the conditional expectation:

$$\bar{\pi}_* \approx \mathbb{E}_{\hat{\mathbb{P}}}(\pi_*|X, Y, X^*) = \int \sigma(\phi^*)\hat{\mathbb{P}}(\phi^*|X, Y, X^*)d\phi^* \tag{19}$$

3 Results

We evaluate the performance and efficiency of the proposed approach on a database of 2042 real images. The database contains 530 images of defective metallic boxes and 1512 images of non-defective ones. First, we will look at the ability of our approach to classify defective and non-defective images. Results demonstrate that the autoencoder can be successfully applied to learning relevant features from different inputs. Combined with the GP classifier, it reaches good performances. Second, we evaluate the proposed method for detecting and localizing defects in images. For both cases, we compare the performances of the autoencoder combined with the GP classifier using the Matérn covariance function [32], and the Newton method to search for the maximum likelihood estimator. We train the autoencoder with 75% of the images, i.e. 1531 images, 1148 with defects and 397 without defects. The rest of the images in the dataset is used for test. In order to remove the test bias, we use 100–fold cross validation, i.e. we run the method 100 times. At each run, we randomly select the training set (75% of the entire dataset) and the remaining 25% are used for test. We then average the performance over the 100 runs. To evaluate the classification quality of the different models, we consider the False Negative (**FN**) and False Positive (**FP**) rates where:

- FN rate corresponds to the number of images labeled as non-defective but classified as defective.
- FP rate corresponds to the number of images labeled as defective but classified as non-defective.

We compare the proposed method with other state-of-the-art methods using the same experimental protocol. We use different image representations combined with two classifiers: (1) the Matlab implementation of K-Nearest Neighbor classifier (KNN). We used the Euclidean distance for finding the nearest neighbors. (2) The Matlab implementation of the Support Vector Machine (SVM) classifier.

Image classification

This section presents the performance of the proposed method when applied to classify images. In our implementation, we used a model which consists of two stacked autoencoders with 50 and 60 hidden layers, respectively. The sparsity penalty λ is set to 10^{-4} for all the experiments. The optimal parameters $\hat{W}$ and $\hat{b}$ were obtained by optimizing the cost using an iterative Newton-based method, initialized using a standard normal distribution.

There are three key steps that can affect the performance of any classification method: (i) The representation of the images and the definition of the feature space. (ii) The analysis of these observations in the feature spaces. (iii) The classifiers used to classify the observations. To illustrate their importance in the application context, we use two classifiers: the K-Nearest Neighbor (KNN) and the Support Vector Machines (SVM) classifiers, and six different features to represent each image:

1. Pre-processed image intensity.
2. Vertical gradient of (a).
3. Binarized version of (b).
4. Histogram of Oriented Gradients (HOG) computed from (a).
5. Coefficients from the decomposition of (a) into a linear combination of the Haar wavelet basis.
6. Gabor descriptor computed from (a) using two directions $\left\{\frac{\pi}{2}, \frac{3\pi}{4}\right\}$

These features are then compared to the method proposed in this article, which can take any input as a feature vector. It then searches for the optimal parameters and features during the training stage, and predicts the correct label for test data. Experimental results show that the KNN classifier performs worse than SVM for all type of features. Compared to KNN, SVM shows better classification performances but remains below the proposed method. Table 1 summarizes the classification performance of these methods in terms of False Negative (FN) and False Positive (FP) rates.

According to Table 1, we note that the proposed method outperforms the other methods regardless of the features used to represent the images. In particular, we observe that the best accuracies are achieved when combining either Haar decomposition and / or HOG descriptors as input with SVM for classification or with our proposed autoencoder model.

Table 1. Classification performance with different features for representing images and different classifiers. The best performances are indicated in bold (the lower the better). These results are obtained using 100-fold cross validation.

Features	Rate	Classification Method		
		KNN	SVM	Autoencoder
Grayscale	FP	29.8%	29.4%	**25%**
	FN	28.5%	27%	**24.8%**
Gradient	FP	26.2%	24.5%	**23.4%**
	FN	29.8%	29.4%	**16.9%**
Binary	FP	25.9%	25%	**21.9%**
	FN	20.1%	15%	**13.2%**
Gabor	FP	38.2%	32%	**26.9%**
	FN	45%	38.3%	**34.8%**
HOG	FP	25%	19.2%	**19%**
	FN	18.9%	13.8%	**10%**
Haar	FP	24.2%	23%	**18.9%**
	FN	17%	14%	**7.9%**

https://doi.org/10.1371/journal.pone.0203192.t001

The ROC curves of Fig 4 show that the proposed method has the most predictive power and generalization capability with a value of 0.86, followed by SVM 0.815 and then KNN 0.794. This indicates that the proposed method succeeds in learning relevant features, reducing dimensionally, and predicting more accurately.

Detection and localization of defects

Finally, we extend the proposed approach to detecting and localizing defects in images. First, to build the training (and testing) data, we asked an expert to manually localize, in each input image, regions that contain defects. We then divided all the images into patches of size 32×32. Patches that contain defects are then treated as negative examples while the remaining are treated as positive examples. All the patches were subject to the same preprocessing step as the one used for the previous experiments. We selected HOG descriptors and Haar coefficients to represent an image since they achieved the best classification accuracy, see Table 1.

To show the effectiveness of our approach, we use the same database for classifying and detecting defective sub-regions. We also compare our method with HOG-SVM and Haar-SVM models, and a pre-trained CNNs such as VGG-16. VGG 16-layer is a deep convolutional neural network pretrained on ImageNet, a large image dataset composed of 1000 classes and 1.3M images [33]. We use the MatConvNet implementation [33] of VGG-16, with an additional fine-tuning step, combined with a softmax classifier.

Table 2 summarizes the defects detection rates for all methods. These results are obtained using 100-fold cross validation using the same setup as the previous experiment. From Table 2, we observe that our approach outperforms the other methods. The ROC curves in Fig 5 confirm that both VGG-softmax and our model reach a good accuracy with a slight advantage for our method.

Finally, Fig 6 shows examples of localized defects on the original test images.

4 Discussion

The new framework proposed in this article, which enables real inspection and classification of defects in metallic boxes, can be generalized for any application that deals with detecting non-standard subregions. The representation part, i.e. feature extraction, was used to illustrate the ability of our framework to capture discriminant information from the input. Other features from the literature could be also used or adapted for the application at hand. The investigation of the best representation is out of the scope of this work. Nevertheless, the features can be selected independently and then used following the same procedure described in this work. One can also use different features selection [34, 35] and similarity learning [36] methods to automatically select the features that achieve the best performance.

It addition to the accuracy of the defect detection results, the proposed framework has at least three advantages compared to the state-of-the-art:

- First, it provides more flexibility compared to pre-trained CNNs: our method can be adapted to the dimension of the input. In fact, it can accept input of any dimension and choose the output dimension by fixing the size of the different layers. This leads to a reduced dimension, which in turn improves the computational efficiency.
- The input can be a vector of any features or a mixture of them. This can lead to a different network architecture but the overall process remains the same.
- The classifier includes a mapping function which includes the non-linearity of data making this framework more general then linear classifiers.

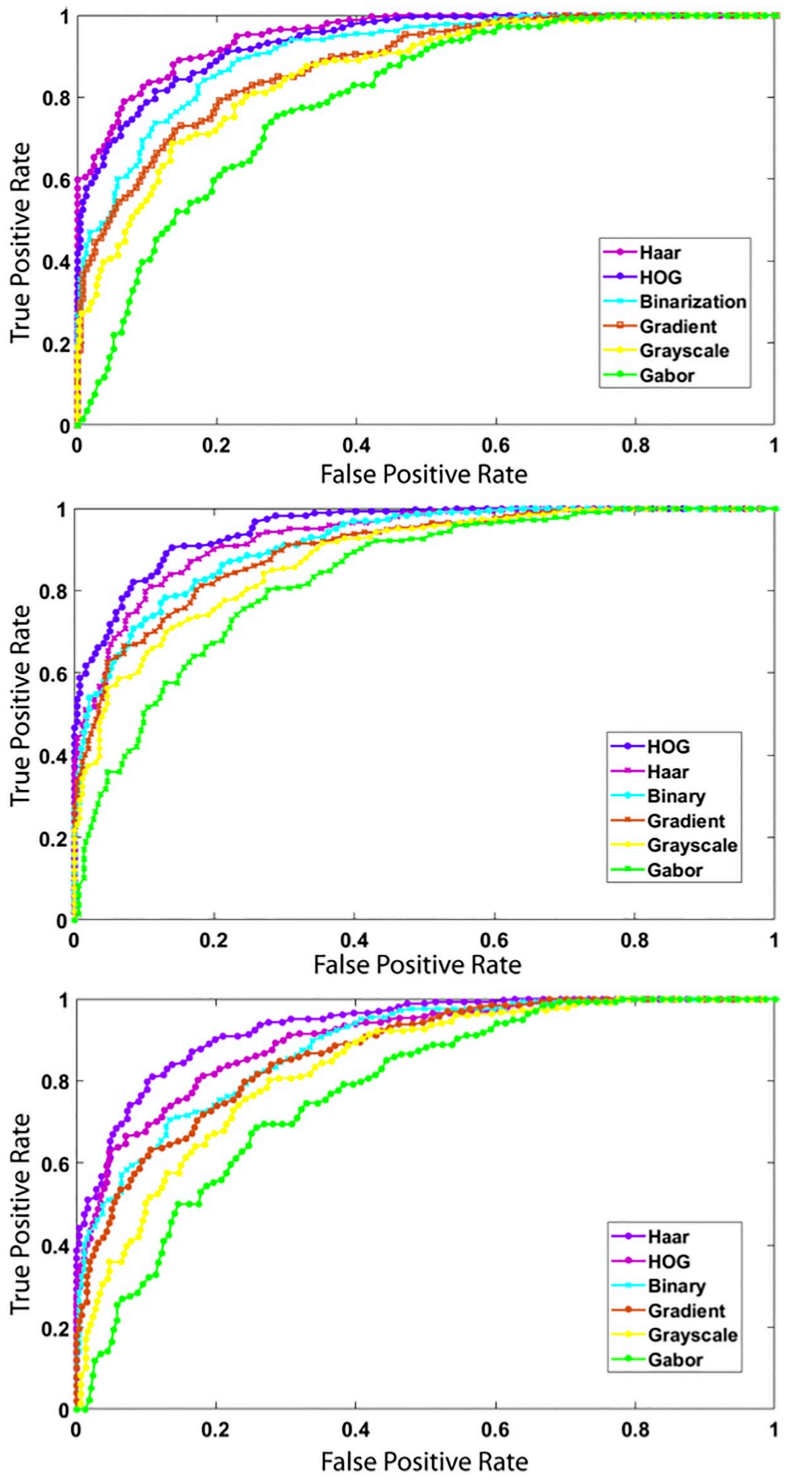


Fig 4. ROC curves of the methods with different images features and different classifiers: the proposed method (top), SVM (middle), and KNN (bottom).

https://doi.org/10.1371/journal.pone.0203192.g004

Table 2. Performance of the methods for detecting defective patches. Rates are obtained using 100-fold cross validation. Here, we provide the average performance and the standard deviation (Std) over the 100 runs.

Method	FP	Std	FN	Std
Haar-SVM	17%	0.98	12.2%	1.10
Haar-KNN	21.8%	1.5	16.5%	1
HOG-SVM	14.5%	1.1	10%	1.2
VGG-softmax	13.02%	1.95	8.6%	1.34
The proposed	**10.6%**	1.4	**5.41%**	1.39

https://doi.org/10.1371/journal.pone.0203192.t002

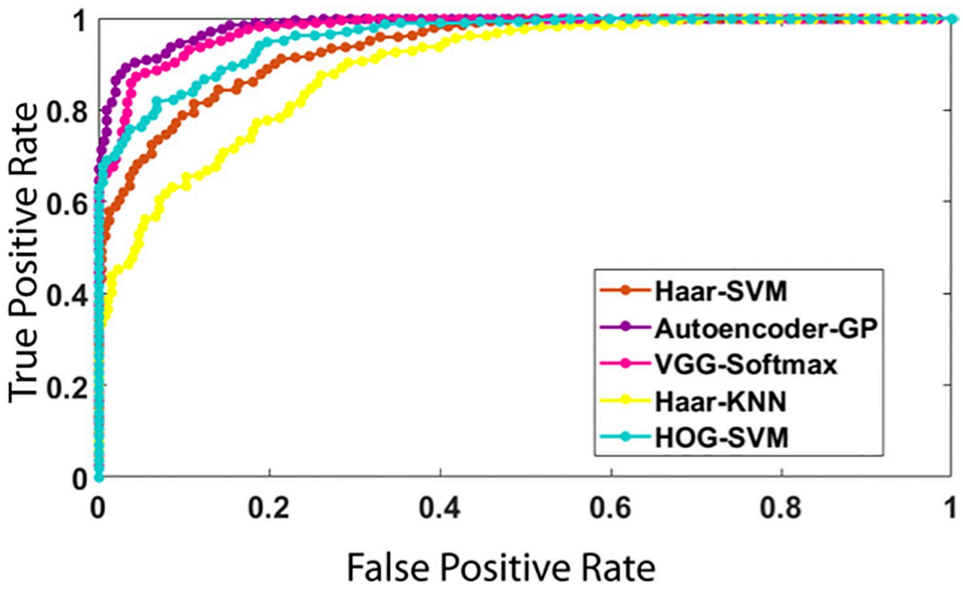


Fig 5. ROC curves of the methods with different features for representing images and different classifiers.

https://doi.org/10.1371/journal.pone.0203192.g005

It should be noted that there is still room for further improvements, especially when it comes to the prediction part where we used a Newton-bas method to search for optimal parameters of the Gaussian process. More sophisticated stochastic tools, such as MCMC, may improve the quality of the estimator.

Fig 7 highlights some failure cases (false positives and false negatives) of the proposed method. This figure shows that it is not always a straightforward decision whether a subregion contains a defect or not (see Fig 7 (top row)). In fact, some defects are very small and that images were manually classified by one expert. This means that it could be hard even for an expert to decide visually what is a defect when it is small or has a different shape (see Fig 7

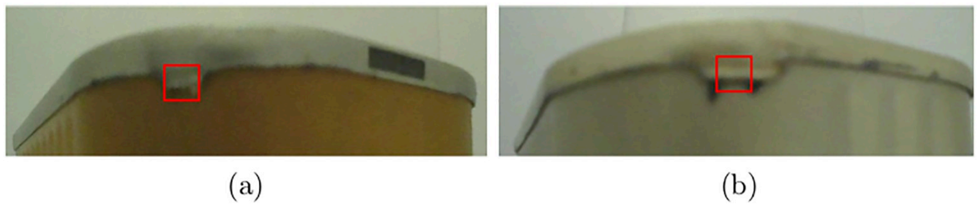


Fig 6. Examples of successful defects detections (a & b) using the proposed method.

https://doi.org/10.1371/journal.pone.0203192.g006

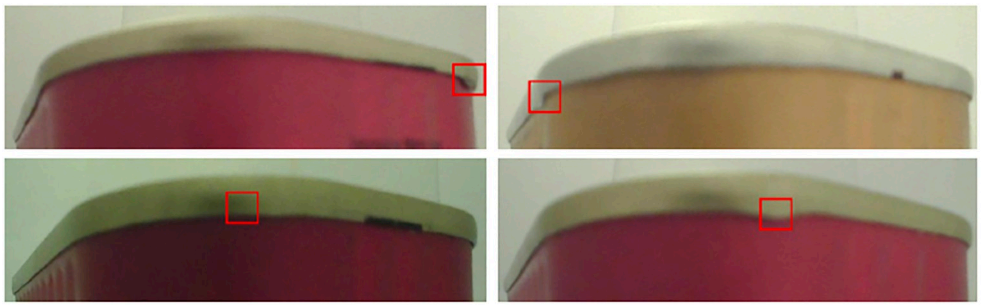


Fig 7. Examples of incorrect detections: False positive (top row) and false negative (bottom row).

https://doi.org/10.1371/journal.pone.0203192.g007

(bottom row)). An extension to this work would be to ask different experts to label images independently and to learn uncertainty at the same time as labels. This will make the framework more complete but will need more complex methodologies to come even close to handling the hard challenges resulting from such formulations.

5 Conclusion

We proposed in this article a new machine learning method for detecting and localizing defects in images of metallic boxes. The proposed method is based on: (1) an autoencoder to automatically learn features from the input, and (2) a Gaussian process classifier. Different image representations were used as an input to the autoencoder and two of them were selected for detection: HOG descriptor and the decomposition in a wavelet basis. To show the effectiveness of our approach, we used the same database for classifying and detecting defective subregions. We have also compared our method with other state-of-the-art techniques, namely the HOG-SVM and the Haar-SVM models, and a pretrained CNNs such as VGG-16. The experimental results demonstrate that the proposed method achieves the best performance. It can be successfully applied to learning relevant features from different inputs and when combined with the GP classifier.

Supporting information

S1 Dataset.
(ZIP)

Author Contributions

Methodology: Oumayma Essid, Hamid Laga, Chafik Samir.

Software: Oumayma Essid.

Supervision: Hamid Laga, Chafik Samir.

Writing – original draft: Oumayma Essid, Hamid Laga, Chafik Samir.

Writing – review & editing: Oumayma Essid, Hamid Laga, Chafik Samir.

References

1. Hajizadeh S, Nunez A, Tax DM. Semi-supervised rail defect detection from imbalanced image data. IFAC-PapersOnLine. 2016; 49(3):78–83. https://doi.org/10.1016/j.ifacol.2016.07.014
2. Chen YJ, Tsai JC, Hsu YC. A real-time surface inspection system for precision steel balls based on machine vision. Measurement Science and Technology. 2016; 21(7):76–82.

3. Golnabi H, Asadpour A. Design and Application of Industrial Machine Vision Systems. Robot Comput-Integr Manuf. 2007; 23:630–637. https://doi.org/10.1016/j.rcim.2007.02.005
4. Pratt WK. Digital Image Processing: PIKS Inside. John Wiley & Sons, Inc.; 2001.
5. Bengio Yoshua. Learning Deep Architectures for AI. Foundations and Trends in Machine Learning. 2009: 1–127.
6. Zhong G, Wang LN, Ling X, Dong J. An overview on data representation learning: From traditional feature learning to recent deep learning. The Journal of Finance and Data Science. 2016; 2:265–278. https://doi.org/10.1016/j.jfds.2017.05.001
7. S. Satorres Martínez and C. Ortega Vázquez and J. Gámez García and J. Gómez Ortega. Quality inspection of machined metal parts using an image fusion technique. Measurement. 2017; 111(Supplement C):374–383.
8. Jian C, Gao J, Ao Y. Automatic surface defect detection for mobile phone screen glass based on machine vision. vol. 52; 2017. p. 348–358.
9. Shanmugamani R, Sadique M, Ramamoorthy B. Detection and clas- sification of surface defects of gun barrels using computer vision and machine learning. Measurement. 2015; 60(Supplement C):222–230. https://doi.org/10.1016/j.measurement.2014.10.009
10. Shen H, Li S, Gu D, Chang H. Bearing defect inspection based on machine vision. Measurement. 2012; 45:719–733. https://doi.org/10.1016/j.measurement.2011.12.018
11. Ngiam J, Khosla A, Kim M, Nam J, Lee H, Ng AY. Multimodal Deep Learning. In: Getoor L, Scheffer T, editors. ICML. Omnipress; 2011. p. 689–696.
12. Tanisik G, Zalluhoglu C, Ikizler-Cinbis N. Facial Descriptors for Human Interaction Recognition In Still Images. CoRR. 2015;abs/1509.05366.
13. Raiko T, Valpola H, Lecun Y. Deep Learning Made Easier by Linear Transformations in Perceptrons. In: Lawrence ND, Girolami M, editors. Proceedings of the Fifteenth International Conference on Artificial Intelligence and Statistics. vol. 22 of Proceedings of Machine Learning Research. PMLR; 2012. p. 924–93
14. Soukup D, Huber-Mörk R. Convolutional Neural Networks for Steel Surface Defect Detection from Photometric Stereo Images. In: Ad- vances in Visual Computing—10th International Symposium, ISVC 2014, Las Vegas, NV, USA, December 8-10, 2014, Proceedings, Part I; 2014. p. 668–677.
15. Gopalakrishnan K, Khaitan SK, Choudhary A, Agrawal A. Deep Convolutional Neural Networks with transfer learning for computer vision-based data-driven pavement distress detection. Construction and Building Materials. 2017; 157:322–330. https://doi.org/10.1016/j.conbuildmat.2017.09.110
16. Bengio Y, Lamblin P, Popovici D, Larochelle H. Greedy layer-wise training of deep networks. In: IN NIPS. MIT Press; 2007.
17. Irsoy O, Alpaydin E. Unsupervised feature extraction with autoencoder trees. Neurocomputing. 2017; 258:63–73. https://doi.org/10.1016/j.neucom.2017.02.075
18. Irsoy O, Alpaydin E. Autoencoder Trees. In: Proceedings of The 7th Asian Conference on Machine Learning, ACML 2015, Hong Kong, November 20-22, 2015.; 2015. p. 378–390.
19. Gonzalez RC, Woods RE. Digital Image Processing (2nd Ed). Prentice Hall; 2002.
20. Rudin LI, Osher S, Fatemi E. Nonlinear Total Variation Based Noise Removal Algorithms. Phys D. 1992; 60:259–268. https://doi.org/10.1016/0167-2789(92)90242-F
21. Ren M, Liao R, Urtasun R, Sinz FH, Zemel RS. Normalizing the Normalizers: Comparing and Extending Network Normalization Schemes. CoRR. 2016; 1.
22. Ng HF. Automatic thresholding for defect detection. Pattern Recognition Letters. 2006; 27(14):1644–1649. https://doi.org/10.1016/j.patrec.2006.03.009
23. Mallat SG. A Theory for Multiresolution Signal Decomposition: The Wavelet Representation. IEEE Trans Pattern Anal Mach In- tell. 1989; 11:674–693. https://doi.org/10.1109/34.192463
24. Sankaran A, Vatsa M, Singh R, Majumdar A. Group sparse autoencoder. Image Vision Computing. 2017; 60:64–74. https://doi.org/10.1016/j.imavis.2017.01.005
25. Ranzato M, Poultney C, Chopra S, Cun YL. Efficient Learning of Sparse Representations with an Energy-Based Model. In: Schölkopf PB, Platt JC, Hoffman T, editors. Advances in Neural Information Processing Systems 19. MIT Press; 2007.
26. Hinton G, Salakhutdinov R. Reducing the Dimensionality of Data with Neural Networks. Science. 2006; 313(5786):504–507. https://doi.org/10.1126/science.1127647 PMID: 16873662
27. Lee JA, Verleysen M. Nonlinear Dimensionality Reduction. Springer Publishing Company, Incorporated; 2007.

28. Dolz J, Betrouni N, Mathilde. Stacking denoising auto-encoders in a deep network to segment the brainstem on MRI in brain cancer patients: A clinical study. Computerized Medical Imaging and Graphics. 2016; 52:8–18. https://doi.org/10.1016/j.compmedimag.2016.03.003 PMID: 27236370

29. Cook RD, Weisberg S. Residuals and influence in regression. Chapman, Hall/CRC, editors. Chapman and Hall, New York—London; 1982.

30. Loh WL. Fixed-domain asymptotics for a subclass of Matérn-type Gaussian random fields. The Annals of Statistics. 2005; 33:2344–2394. https://doi.org/10.1214/009053605000000516

31. Bachoc F. Asymptotic analysis of the role of spatial sampling for covariance parameter estimation of Gaussian processes. Journal of Multivariate Analysis. 2014; 125:1–35. https://doi.org/10.1016/j.jmva.2013.11.015

32. Rasmussen CE, Williams CKI. Gaussian Processes for Machine Learn- ing. MIT Press; 2006

33. Vedaldi A, Lenc K. MatConvNet—Convolutional Neural Networks for MATLAB. In: Proceeding of the ACM Int. Conf. on Multimedia; 2015

34. Zhu P, Zhu W, Hu Q, Zhang C, Zuo W. Subspace clustering guided unsupervised feature selection. Pattern Recognition. 2017; 66:364–374. https://doi.org/10.1016/j.patcog.2017.01.016

35. Zhu P, Xu Q, Hu Q, Zhang C, Zhao H. Multi-label feature selection with missing labels. Pattern Recognition. 2018; 74:488–502. https://doi.org/10.1016/j.patcog.2017.09.036

36. Laga H, Nakajima M. Supervised learning of similarity measures for content-based 3D model retrieval. In: Large-Scale Knowledge Resources. Construction and Application. Springer, Berlin, Heidelberg; 2008. p. 210–225.