Int J Adv Manuf Technol
DOI 10.1007/s00170-017-0882-0

ORIGINAL ARTICLE

A fast and robust convolutional neural network-based defect detection model in product quality control

Tian Wang[1] · Yang Chen[1] · Meina Qiao[1] · Hichem Snoussi[2]

Received: 15 June 2017 / Accepted: 24 July 2017

Abstract The fast and robust automated quality visual inspection has received increasing attention in the product quality control for production efficiency. To effectively detect defects in products, many methods focus on the handcrafted optical features. However, these methods tend to only work well under specified conditions and have many requirements for the input. So the work in this paper targets on building a deep model to solve this problem. The elaborately designed deep convolutional neural networks (CNN) proposed by us can automatically extract powerful features with less prior knowledge about the images for defect detection, while at the same time is robust to noise. We experimentally evaluate this CNN model on a benchmark dataset and achieve a fast detection result with a high accuracy, surpassing the state-of-the-art methods.

Keywords Product quality control · Defect detection · Convolutional neural networks

1 Introduction

As an important part of the product quality control, the quality visual inspection of products is gaining more and more attention in the industrial manufacture [1–4]. It aims to ensure the product quality by detecting defects by visual means. In the production process, the low product quality will do no good but harm to any of the participants. On the one hand, the quality of products will affect the production efficiency, since products with poor quality will crimp the sales and they are only a waste of raw materials and costs money. On the other hand, product quality relates to the products in the market share and the credibility of the factory. Only products with high-quality can obtain the long-term occupation of the market. However, in the process of quality visual inspection, many factories still use artificial methods to do defect detection, which rely heavily on the manpower and consume much financial resources. Since a person's own energy is limited and the human inspection job is trivial, workers with long-time continuous work may reduce production efficiency because of fatigue, and this will lead to the problem of product quality testing and heavy economic losses brought by the human error. To this end, it is urgent to bring automated defect detection into the production process.

With the rapid development of computer technology and the expansion of its application fields [5, 6], computer vision has been successfully applied in the quality inspection of various industrial products, including steel slabs, glass products, fabrics, polycrystalline solar wafers, and so on. These methods tend to design different algorithms to extract image features based on actual defect detecion conditions. For the steel slab, a vision-based automatic detection technique was proposed in [1] to detect steel surface defects, like cracks and scratches covered by industrial liquids in the cold rolling process. In this work only one parameter was needed to be preset at first, and once selected properly, this parameter was robust to defective images of different types. For glass products, the PCA-based defect inspection system for mobile phone cover glasses in [2] was proposed, which can not only recognize several defect types such as

✉ Tian Wang
wangtian@buaa.edu.cn

[1] School of Automation Science and Electrical Engineering, Beihang University, Beijing, China

[2] Institut Charles Delaunay-LM2S-UMR STMR 6281 CNRS, University of Technology of Troyes, Troyes, France

Published online: 15 August 2017

scratch, crack, deformation, edge broken, and angle cutting simultaneously, but can also recognize the defect type robustly different from others based on the defect shapes or structures. Another kind of glass manufacturing was also introduced in paper [3]. It presented an intelligent system based on computer vision for the automatic inspection of 2 kinds of defects in glass products, one was the critical defect in glass cups for food packaging and domestic use called glass sparkle or fragment of glass, and the other was the defect called deformation in plates. For fabrics defect detection [4], most algorithms employed Gabor filters and Gabor wavelet filters [7–9]. In the work of [10], an embedded machine vision system using Gabor filters and Pulse Coupled Neural Network (PCNN) was developed to identify defects of warp-knitted fabrics automatically, which consisted of image enhancement implemented by Gabor filtering with optimal parameters to make the defects more obvious, and image segmentation achieved by a parameter adaptive PCNN layer by layer. Different from the Gabor filter, the paper [11] proposed a regularity measurement for defect detection in non-textured and homogeneously textured images using principal component analysis (PCA), which is an orthogonal transformation to transform linearly and non-linearly the correlation of the source variables into a subspace in which the variables are not correlated. It is widely used in feature extraction and data compression and usually used for data preprocessing in the defect detection. For example, the paper [12] characterized an approach to defect isolation for the multi-variable process by wavelet PCA, which was implemented on a statistic system and the Tennessee Eastman process to obtain satisfactory performance.

The widely used defect detection methods have many restrictions like sensitive to light conditions, needed to be redesigned when new problems arise and so on. With the development of big data [13, 14] and the enhancement of computer computing ability, deep learning strategies have been successfully applied in various fields. The deep convolutioal neural networks (CNN), a representative of them, can learn a hierarchy of features from the raw image input by automatically update the filters during training on massive amounts of training data. For defect detecion, the paper [15] proposed a novel deep CNN architecture to detect defects, which taked all types of defect free and defective samples together as the input. This model is a 12-class classifier: 6 defect free classes and 6 defective classes.

In our work, we propose a elaborately designed joint detection CNN architecture to address the defect detection problem. This network can automatically extract useful robust features and it works in a two-fold procedure that for an image sample, firstly the sample class is decided based on its background texture information, and then whether it contains defective regions or not is indicated. Our model works very fast on the validation stage and achieve a satisfactory correct defect detection rate.

2 Methodology

We use the CNN architecture for defect detection in our work. And this part is organized as follows. We firstly provide some theoretical background information about the CNN network needed for the model in Section 2.1. Then, we briefly describe the database used in our work in Section 2.2. Section 2.3 centers on the proposed CNN framework of our model. And Section 2.4 demonstrates the critical training details.

2.1 Convolutional neural networks

The convolutional neural networks (CNN) is one typical type of the artificial neural networks (ANN). The CNN can automatically learn a hierarchy of features from the input image matrices, which prove to be better than those handcrafted features extracted by carefully designed complex algorithms. In the recent years, the CNN model has made a major breakthrough in computer vision and is widely used in a variety of applications such as image classification [16], image segmentation [17], object tracking [18], and so on. Due to the delicate design of the CNN architecture including weight parameter sharing and pooling operations, the CNN is easier to train with fewer connections and less parameters, and the learnt features tend to be shift invarient.

A typical CNN architecture consists of several nested convolutional and pooling layers followed by fully connected layers at the end. A simplified presentation of this kind of network can be [Input - Conv - ReLU - Pool - FC]:

- Input: The input of the CNN tends to be the 3-channel color image or 1-channel gray image matrices containing the intensity values at each position.
- Conv: The convolutional layers apply a set of filters each of which is connected to only a small regions of the output of last layer called the receptive field. And the filters are usually learnable matrices during training with samll size like 3×3 or 5×5. The parameter sharing scheme is applied in the convolution operations that one filter is convolved over the whole image input across the spatial dimensions to extract one feature.
- ReLU: ReLU (Rectified Linear Units) is the most widely used activation function by adding non-linear transformations to the output response of the convolutional or fully connected layers. The formula of this function is $f(x) = max(0, x)$. As one of the several keys to the recent success of the deep networks [16], ReLU can effectively prevent the gradients from

saturating, expedite convergence of the training procedure while at the same time keep the original value to the most extent, which proves to be experimental better than conventional sigmoid-like activation functions [16, 19].

– Pool: The pooling layer performs a form of non-linear down-sampling along the both spatial dimensions, leading to reduced spatial size of the output. It aims to reduce the amount of the network parameters and the computation cost. The pooling layer is common to be placed between two successive convolutional layers and the most common pooling strategy is max pooling, which outputs the max value from the neighborhood of the input feature map.
– FC: The fully connected layers are the last part of the neural networks. All the neurons in the fully connected layers are connected to all the units of last layer. And the last fully connected layer generates the output of the whole network with K neurons, the same number as the input labels. Each value of the K-dimensional output represents the probability of the corresponding label with the help of the softmax function:

$$p(z_i) = \frac{exp(z_i)}{\sum_{i=1}^{K} exp(z_i)}, \tag{1}$$

where z_i denotes the ith value computed by the second last layer and $p(z_i)$ denotes the prediction probability.

After all these layers stack together to form a complete CNN, the input is fed forward into the network for decision making. And the hyperparameters are updated by the back propagation algorithm.

2.2 The dataset

A benchmark dataset [20] is proposed by the competition called "Weakly supervised learning for industrial optical inspection" provided by DAGM (German Association for Pattern Recognition) and GNNS (German Chapter of the European Neural Network Society). And we evaluate our method on this dataset, which is referred to as the DAGM dataset in the following. The DAGM dataset which is shown in Fig. 1 contains image samples of 6 classes with size 512 × 512 pixel. Each class consists of 1000 defect free images and 150 defective ones with only one labeled defect region each on the texture background. This dataset is highly challenging mainly for two reasons:

– the defective background texture in the same class varies a lot
– some of the defect regions are very small and some are very similar to the backgroud texture

In our experiments, we randomly choose 70% of the data as the training set and the rest 30% as the validation set. This makes 700/106 (defect free/ defective) samples for training and 300/44 for validation in each class.

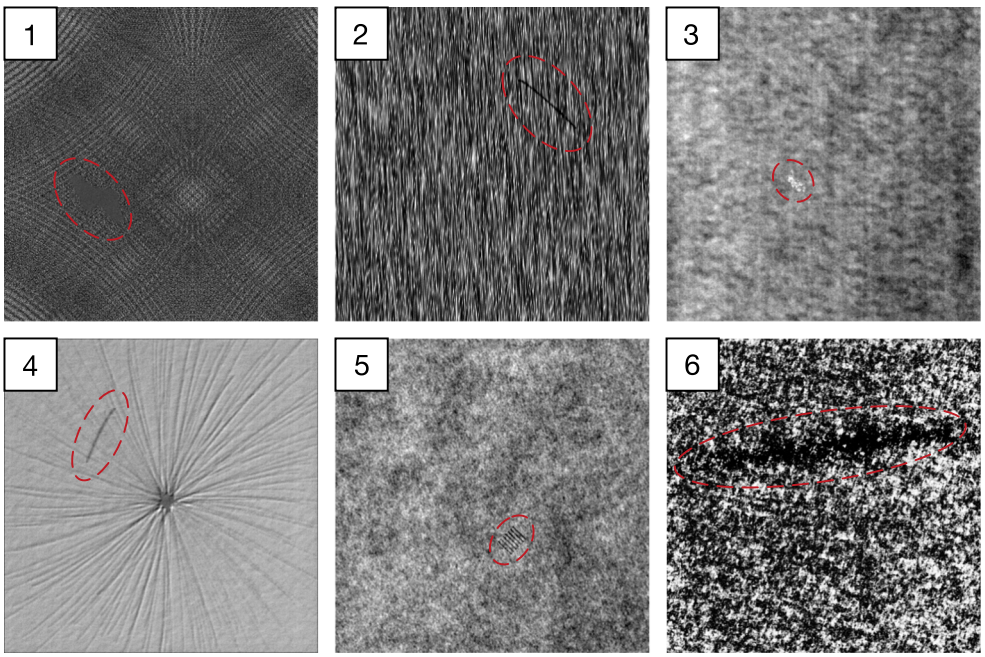

Fig. 1 The original image samples from 6 classes, each of them differs from each other on the background texture. The defective regions are marked out by a surrounding ellipse, shown in *red*

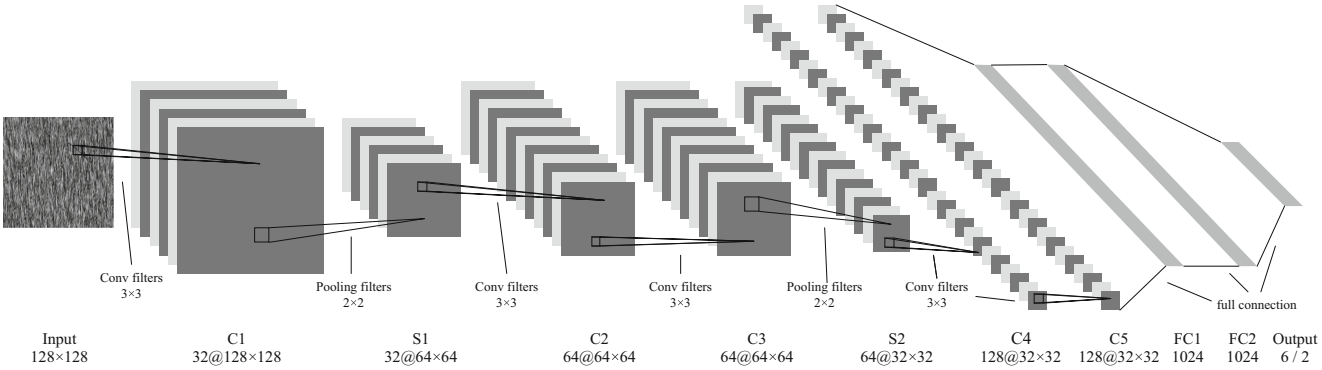

Fig. 2 The architecture of our 11-layer CNN networks. The first global frame classification part takes the down-sampled images of size 128 × 128 pixel as input and outputs 6 neurons, each of which represents the corresponding class membership probability. The second sub-frame detecion part takes extracted mini blocks of the same size with the first part and only outputs 2 neurons indicating whether the block is defective or not

2.3 Model implementation

Considering that the defect detection is the problem of image processing in essence, the CNN network is applied in our work. To detect the defect areas in the images and label them correctly, the implementation of our model is twofold. We design a joint detection CNN architecure which contains two major parts: the global frame classification part and the sub-frame detection part. The global frame classification part learns to classify the image samples into the correct class based on their background texture features. The sub-frame detection part is developed to decide whether each of the samples contains defective regions or not based on the output of the first part. The two parts are quite similar in architecture and they are strung together for the defect detection forming the whole network.

As shown in Fig. 2, the global frame classification part aimed for image class classification consists of 11 layers. For shorthand notation, the architecture can be denoted by $C(32,\ 3,\ 3)$-$S(2,\ 2,\ 2)$-$C(64,\ 3,\ 3)$-$C(64,\ 3,\ 3)$-$S(2,\ 2,\ 2)$-$C(128,\ 3,\ 3)$-$C(128,\ 3,\ 3)$-$S(2,\ 2,\ 2)$-$FC(1024)$-$FC(1024)$-$FC(6)$ where $C(n, 3, 3)$ represents a convolutional layer with n filters of kernel size 3×3, $S(2, 2, 2)$ represents a pooling layer with a subsampling factor of 2×2 by stride 2 in both dimensions, and $FC(n)$ represents a fully-connected layer with n neurons. To reduce the model's computation cost, the whole 512×512 pixel image is down-sampled to size 128×128 pixel at first and then gets fed into the network as the input. We also apply the padding strategy that pads zeros around the borders of the feature maps after each convolution leading to unchanged output shape in each channel. And the pooling strategy adopted in all the pooling layers is max-pooling, which is robust to small distortions. Finally, the last fully connected layer generates the output vectors each containing 6 units corresponding to the number of the classes of the image samples, and the 6 values represent the estimation of each image's class membership probability separately via the softmax regression function.

The second sub-frame detection part is quite similar to the first part in architecture except for the last fully connected layer, which outputs a vector of 2 values indicating whether the input image is defective or not. Different form the first part, the second part takes image blocks of size 128×128 pixel extracted from the original 512×512 pixel images as the input. The extraction is performed by the sliding-window method depicted in Fig. 3. The sliding window has a size of 128×128 pixel and moves along the rows and columns over the whole image with a 64 pixel stride. In

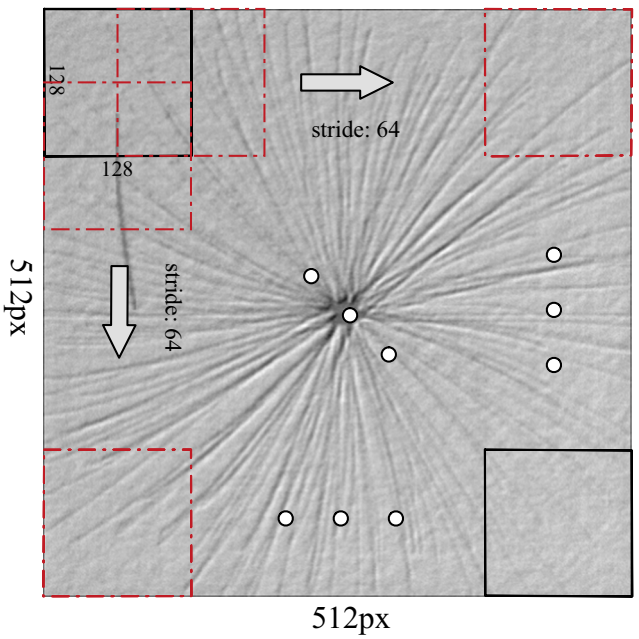

Fig. 3 The sliding-window method we use to extract blocks from the original images. 49 blocks with size 128×128 pixel are extracted from each single image sample

this way, we can extract 49 small blocks from one original image. Thus, we need to train 6 such sub-networks, each of which takes the defect free and defective blocks from one class as input.

The global frame classification part works in conjunction with the sub-frame detection part for the purpose of defect detection. Given an image, the first part decides which class this sample belongs to. Then, this image is divided into 49 blocks and they are fed into one of the sub-networks belonging to the second part according to the output of the first part. If any of the blocks is detected as defective, then the whole image is labelled defective.

2.4 Training details

After the train-validation split, the training set has a small sample size. However, there is no observation of overfitting in the training process of the first global frame classification part. The major reason is that the background texture varies a lot from each other in the 6 classes so that it is easy for the first part to distinguish among them. Actually, in the first sub-network, the evenly distributed samples have the sample size enough in quantity to prevent from overfitting. In the second sub-network, the defective images only account for a small proportion (about 13%) in each class and only a small region in each defective image is labelled defective. Thus, to prevent the second sub-network sets from overfitting, we apply the effective data augmentation strategy on the training data. For the defective images in the training set of each class, we firstly apply the sampling method which manually extracts 22 defective blocks with size 128×128 pixel from each image. Next, we augment these blocks by a factor of 8 via linear transformations including rotations and mirrorings. For the defect free samples, to alleviate the imbalance of sample size between the defect free and the defective blocks, we perform the sliding-window method with 128×128 pixel sliding window size and 64 pixel strides along the x and y axis, which enlarges the number of image blocks by a factor of 49. In this way, the training set for the second CNN now has 34300 defect free blocks and 18656 defective blocks for each class.

We select the cross-entropy function as the loss function of our model. And during the training process, the stochastic gradient descent with mini-batches of 50 samples is applied to update the weight parameters. We also incorporate the momentum and learning rate decay into the stochastic gradient descent optimizer and the updating rules of the weights in each iteration are as follows:

$$v_{i+1} = \mu \cdot v_i - lr \cdot \nabla g, \tag{2}$$

$$w_{i+1} = w_i + v_{i+1}, \tag{3}$$

$$lr = lr \cdot \frac{1}{1 + d \cdot i}, \tag{4}$$

where i is the iteration index, w is the weight hyperparameter, μ is the momentum coefficient, v is the current velocity vector, lr is the learning rate, d is the decay parameter of the learning rate and ∇g is the average value of gradients with respect to w over the mini-batch at each iteration. In our experiments, we set μ and d to 0.9 and 0.012, with which we observe faster convergence speed and less training errors.

The weight parameters in each layer are initialized from a truncated random normal distribution subject to $N \sim (0, \frac{2}{n})$, where n denotes the number of connections between two layers. And we initialize the bias value to 0 for every layer and choose the ReLU activation function according to the work of He et al. [19].

Table 1 The overall detection results of our model and the comparison to the existing methods

Class	Our model	12-class CNN [15]	Statistical features [23]	SIFT and ANN [24]	Weibull [25]
TPR (%)					
1	100	100	99.4	98.9	87.0
2	100	100	94.3	95.7	–
3	100	95.5	99.5	98.5	99.8
4	100	100	92.5	–	–
5	99.7	98.8	96.9	98.2	97.2
6	100	100	100	99.8	94.9
FNR (%)					
1	100	100	99.7	100	98.0
2	100	97.3	80.0	91.3	–
3	100	100	100	100	100
4	93.2	98.7	96.1	–	–
5	100	100	96.1	100	100
6	100	99.5	96.1	100	100
Average accuracy (%)					
	99.8	99.2	95.9	98.2	97.1

Furthermore, we introduce the L2 regularization which adds a weight decay term to the loss function to penalize the large weights during training and avoid overfitting. And the regularization coefficient λ is set to 5×10^{-5} in our experiments. Apart from that, during the training stage we apply the drop-out strategy [21] with probability 0.5 to the next to last and the third from last fully-connected layers which also helps to avoid overfitting. To expedite the training procedure, the batch normalization [22] which can address the internal convariate shift problem is also adopted.

3 Results

We validate our approach on the DAGM defect dataset following the two-fold procedure and achieve the overall detection accuracy of 99.8%, outperforming the published best performance by Daniel et al. [15] which designs a 12-class CNN network. The results and comparisons are summarized in Table 1. We report the true positive rate (TPR) of defect free images correctly classified as defect free and the ture negative rate (TNR) which indicates the percentage of defective images classified as defective.

The results show that our model is able to distinguish between the defect free and defective images. And it can be noticed that except the FNR in class 4, our model outperforms all the existing methods, which proves the effectiveness of our model.

Our model is trained on one NVIDIA GTX1080 8GB GPU for roughly 8 h. And during validation stage, the model is able to detect 27 images per second, which guarantees real-time detection with high accuracy.

4 Conclusions

In this work, we propose a twofold joint detection CNN network to automatically extract powerful image features for defect detection. We evaluate this method on the DAGM dataset consisting of 6 different image categories, each of which differs from others on the background texture. The experiments show that our model is able to classify the image sample into the correct image class and indicate wheter it contains defective regions or not. This model achieves a high accuracy of 99.8% of correct defect detecion rate on the DAGM dataset, outperforming the state-of-the-art methods while at the same time keeps a high processing speed which guarantees the real-time detection.

Acknowledgements This work is partially supported by the ANR AutoFerm project and the Platform CAPSEC funded by Région Champagne-Ardenne and FEDER, the Fundamental Research Funds for the Central Universities (YWF-14-RSC-102), the National Natural Science Foundation of China (U1435220, 61503017), the Aeronautical Science Foundation of China (2016ZC51022).

References

1. Zhao YJ, Yan YH, Song KC (2017) Vision-based automatic detection of steel surface defects in the cold rolling process: considering the influence of industrial liquids and surface textures. Int J Adv Manuf Technol 90(5-8):1665–1678
2. Li D, Liang LQ, Zhang WJ (2014) Defect inspection and extraction of the mobile phone cover glass based on the principal components analysis. Int J Adv Manuf Technol 73(9-12):1605–1614
3. Cabral JDD, de Araújo SA (2015) An intelligent vision system for detecting defects in glass products for packaging and domestic use. Int J Adv Manuf Technol 77(1-4):485–494
4. Ngan HY, Pang GK, Yung NH (2011) Automated fabric defect detection—a review. Image Vis Comput 29(7):442–458
5. Tao F, Cheng Y, Da Xu L, Zhang L, Li BH (2014) Cciot-cmfg: cloud computing and internet of things-based cloud manufacturing service system. IEEE Trans Ind Inf 10(2):1435–1442
6. Tao F, Zuo Y, Da Xu L, Zhang L (2014) Iot-based intelligent perception and access of manufacturing resource toward cloud manufacturing. IEEE Trans Ind Inf 10(2):1547–1557
7. Tolba A, Raafat H (2015) Multiscale image quality measures for defect detection in thin films. Int J Adv Manuf Technol 79(1-4):113–122
8. Tolba A, Atwan A, Amanneddine N, Mutawa A, Khan H (2010) Defect detection in flat surface products using log-gabor filters. Intern J Hybrid Intell Syst 7(3):187–201
9. Sajid T, Ali B (2012) Fabric defect detection in textile images using gabor filter. IOSR J Electric Electron Eng 3(2):33–38
10. Li Y, Zhang C (2016) Automated vision system for fabric defect inspection using gabor filters and pcnn. SpringerPlus 5(1):765
11. Tsai DM, Chen MC, Li WC, Chiu WY (2012) A fast regularity measure for surface defect detection. Mach Vis Appl 23(5):869–886
12. Chaouch H, Najeh T, Nabli L (2017) Multi-variable process data compression and defect isolation using wavelet pca and genetic algorithm. Int J Adv Manuf Technol 91(1-4):869–878
13. Li J, Tao F, Cheng Y, Zhao L (2015) Big data in product lifecycle management. Int J Adv Manuf Technol 81(1-4):667–684
14. Tao F, Cheng J, Qi Q, Zhang M, Zhang H, Sui F (2017) Digital twin-driven product design, manufacturing and service with big data. Int J Adv Manuf Technol 4:1–14
15. Weimer D, Scholz-Reiter B, Shpitalni M (2016) Design of deep convolutional neural network architectures for automated feature extraction in industrial inspection. CIRP Ann Manuf Technol 65(1):417–420
16. Krizhevsky A, Sutskever I, Hinton GE (2012) Imagenet classification with deep convolutional neural networks. In: Advances in neural information processing systems, pp 1097–1105
17. Ren S, He K, Girshick R, Sun J (2015) Faster r-cnn: towards real-time object detection with region proposal networks. In: Advances in neural information processing systems, pp 91–99
18. Redmon J, Divvala S, Girshick R, Farhadi A (2016) You only look once: unified, real-time object detection. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 779–788
19. He K, Zhang X, Ren S, Sun J (2015) Delving deep into rectifiers: Surpassing human-level performance on imagenet classification.

In: Proceedings of the IEEE international conference on computer vision, pp 1026–1034
20. Jager M, Knoll C, Hamprecht FA (2008) Weakly supervised learning of a classifier for unusual event detection. IEEE Trans Image Process 17(9):1700–1708
21. Srivastava N, Hinton G, Krizhevsky A, Sutskever I, Salakhutdinov R (2014) Dropout: a simple way to prevent neural networks from overfitting. J Mach Learn Res 15(1):1929–1958
22. Ioffe S, Szegedy C (2015) Batch normalization: accelerating deep network training by reducing internal covariate shift. arXiv preprint arXiv:1502.03167
23. Jiang X, Scott P, Whitehouse D (2008) Wavelets and their applications for surface metrology. CIRP Ann Manuf Technol 57(1):555–558
24. Siebel NT, Sommer G (2008) Learning defect classifiers for visual inspection images by neuro-evolution using weakly labelled training data. In: IEEE congress on evolutionary computation, 2008. CEC 2008. (IEEE World congress on computational intelligence). IEEE, pp 3925–3931
25. Timm F, Barth E (2011) Non-parametric texture defect detection using weibull features. In: IS&T/SPIE electronic imaging. International Society for Optics and Photonics, pp 78,770j–78,770j