CIRP Annals - Manufacturing Technology 65 (2016) 417–420

Contents lists available at ScienceDirect

CIRP Annals - Manufacturing Technology

journal homepage: http://ees.elsevier.com/cirp/default.asp

Design of deep convolutional neural network architectures for automated feature extraction in industrial inspection

Daniel Weimer [a,c], Bernd Scholz-Reiter (1)[b,*], Moshe Shpitalni (1)[c]

[a] BIBA – Bremer Institut für Produktion und Logistik GmbH, University of Bremen, Bremen, Germany
[b] University of Bremen, Bremen, Germany
[c] Department of Mechanical Engineering, Technion – Israel Institute of Technology, Haifa, Israel

ARTICLE INFO

Keywords:
Quality assurance
Artificial intelligence
Deep machine learning

ABSTRACT

Fast and reliable industrial inspection is a main challenge in manufacturing scenarios. However, the defect detection performance is heavily dependent on manually defined features for defect representation. In this contribution, we investigate a new paradigm from machine learning, namely deep machine learning by examining design configurations of deep Convolutional Neural Networks (CNN) and the impact of different hyper-parameter settings towards the accuracy of defect detection results. In contrast to manually designed image processing solutions, deep CNN automatically generate powerful features by hierarchical learning strategies from massive amounts of training data with a minimum of human interaction or expert process knowledge. An application of the proposed method demonstrates excellent defect detection results with low false alarm rates.

© 2016 CIRP.

1. Introduction

Optical Quality Control (OQC) and machine vision are vital processes in manufacturing to satisfy customer requirements, which in return will establish sales continuity of the company. To have satisfactory products, one essential step in OQC is to ensure that the product is visually free of imperfections or defects, among other requirements. In many manufacturing industries, human inspection is still a critical element in the process. Although visual inspection is a trivial task for humans to solve, in fast-paced modern industry such a task can be highly repetitive and mind-numbing, which potentially leads to human error due to fatigue. However reliable this may be, OQC performance uncertainty due to human error could be very expensive for the company. Moreover, with increasing production volume, performance of human-based OQC does not scale well as it is constrained with low inspection frequency, not to mention the cost. For these reasons, automated defect detection naturally emerges as a solution to this problem.

The aim of the defect detection process is to segment a possible defective area from the background and classify it in predefined defect categories. In a controlled environment, characterized by stable lighting conditions, simple thresholding techniques are often sufficient to segment defects from the background. However, such methods are no longer applicable when dealing with surfaces with strong or complex textures or noisy sensor data. In these much more challenging applications, more elaborate methods are required to ensure stable and reliable defect detection results. As described by Xie [1] and Neogi et al. [2], existing methods can roughly be divided into four main categories: statistical, structural, filter based, and model based. In industrial defect detection systems, intensive studies have been performed in order to hand-craft the optimum feature representation of the data for the given problem. Pernkopf and O'Leary [3] for example, performed a comparative study on feature selection from an exhaustive list of different feature encodings. Jiang et al. demonstrated the power of filter-based methods, namely three different wavelet formulations, for surface analysis [4]. While these methods usually yield satisfactory results for known problems, they cannot always be applied to new or different problems set because each problem has its own characteristics that only responds to certain kind of feature extractor.

Thus, it is common practice in industrial environments that new feature has to be manually engineered when a new set of problems arises. Additionally, surface defects can occur in arbitrary size, shape and orientation. Therefore, standard feature descriptors for defect description often lead to insufficient classification results [5].

Given these considerations, this contribution investigates an alternative approach, namely Convolutional Neural Networks (CNN), in order to overcome the difficulties of redefining manually a specific feature representation for every new inspection problem. A CNN consists of an arbitrary initialized set of filters, where the goal of a supervised learning procedure is to learn the best filters for a given problem.

CNN automatically generate meaningful features for a specific task in an evolutionary way directly from huge amounts of raw data with minimal human interaction. In this contribution, we

* Corresponding author.
E-mail address: bsr@biba.uni-bremen.de (B. Scholz-Reiter).

http://dx.doi.org/10.1016/j.cirp.2016.04.072
0007-8506/© 2016 CIRP.

investigate the design of CNN architectures towards a robust and generalizable method for industrial inspection and demonstrate the potential to scale well with respect to big data challenges in manufacturing.

2. Theoretical background

2.1. General architecture

Artificial Neural Networks (ANN) in general are a collection of trainable mathematical units (neurons), that collectively learn non-linear functions and already have demonstrated practical usage in different manufacturing application domains like process planning [6] or production control [7].

In contrast to fully connected ANN, where each neuron from layer (l) is connected to all neurons in the previous layer ($l-1$), CNN is a specific type of feed-forward ANN and neurons are connected locally. This design is inspired by the working principle of the visual cortex in the brain [8]. The local connection is realized by a set of arbitrarily initialized filters, so that the main process step of a CNN is a convolution of the filters with a given input image. The architecture is composed of multiple feature extraction stages. Each stage consists mainly of three basic operations: convolution, non-linear neuron activation and feature pooling. Fig. 1 illustrates the basic architecture of a deep CNN. A CNN is denoted as *deep*, when multiple layers of feature extraction stages are connected together [9,10].

With an image as the input, feature map C1 would be the result of a filter convoluted throughout the whole image, followed by a non-linear activation function. The Rectified Linear Unit (ReLU) is used as neuron activation function, as it performs best with respect to runtime and generalization error [11]. The non-linear ReLU neuron activation function follows the formulation $f(z) = max(0,z)$ for each input feature z.

Feature pooling typically involving max or average operation, is then performed to obtain the compressed feature representation S1 by grouping pixel in feature map C1 in a 2×2 neighbourhood. These three layers can be stacked together which results in a hierarchical and evolutionary development of raw pixel data towards powerful feature representations. After a sequence of convolution, ReLU and pooling stages, the extracted features in S2 are forwarded into an ANN architecture for final decision making.

2.2. Training of deep architectures

The crucial processing step when applying deep architectures is the training procedure as a deep architecture consists of millions of parameters which have to be learned during training. Prior to work from Hinton and Salakhutdinov, adding more hidden layer in a neural architecture had no significant impact with respect to the output result when applying *backpropagation of errors* for neuron weight training [12]. The so called *vanishing gradient problem* led to a poor adjustment of neuron weights in the early layers and insufficient generalization performance. To overcome these limitations, the deep network was divided in autoencoder stages and finally stacked together and fine-tuned as a complete network [12]. New strategies for training deep architectures allow the implementation of backpropagation in combination with additional techniques also for deep architectures and will be the basic training procedure for this contribution.

Training of a deep CNN is realized as a supervised training procedure using training representations of the form $(x^{(t)}, y^{(t)})$. Let $\Theta = \{w_1, b_1, \ldots, w_{m+1}, b_{m+1}\}$ be the set of all parameters (neuron weights and biases) from all the layers in a given CNN architecture, while $x^{(t)}$ and $y^{(t)}$ denotes the tth input and its label respectively. From here we frame the problem of learning/training a model as a problem of finding the parameter configuration that minimizes the following objective:

$$\Theta^* = \underset{\Theta}{\operatorname{argmin}} \frac{1}{T} \sum_t L(f_\Theta(x^{(t)}, y^{(t)}) + \lambda \Omega(\Theta)) \qquad (1)$$

The matrix Θ^* is the resulting set of parameters of a CNN minimizing the average loss function $L()$, which compares the network output $f(x)$ which is parameterized with Θ with the correct label $y^{(t)}$. The parameter λ is the learning rate and $\Omega()$ a regularization term which penalize high neuron weights to prevent overfitting. This methodology of training a CNN is a standard backpropagation approach, as the resulting error of the forward pass is propagated back in the network to adapt the neuron weight. Stochastic gradient descent (SGD) is used to optimize the non-convex optimization problem in Eq. (1). SGD allows the training of CNN architectures with small batches of training data, which can be understood as noisy sub-samples of the complete training set.

An additional strategy during training to prevent overfitting of the large network architecture is dropout [13]. The basic concept is to randomly (with a specific probability) drop neurons and their weight respectively from the neural network during training. An additional effect of dropout is a more effective training and especially prediction process, which is important for real world applications in dynamic and fast manufacturing processes. The next section introduces the specific architecture for deep CNN in industrial inspection.

3. Design of deep CNN for industrial inspection

A drawback of ANN in general is the huge amount of hyper-parameter, which have to be set properly in order to achieve best detection performance. In contrast to neuron parameters which are learned during training, hyper-parameters are manually pre-selected and affect the architecture of the artificial network. A selection of main hyper-parameters are number of neurons per layer, number of layers, definition of neuron activation function as well as dropout or learning rate. In case of CNN, additional hyper-parameters like filter size occur. We refer to the selection of hyper-parameters as the design of a deep CNN.

Table 1 describes the CNN configurations for visual defect detection under investigation. The network is divided into three

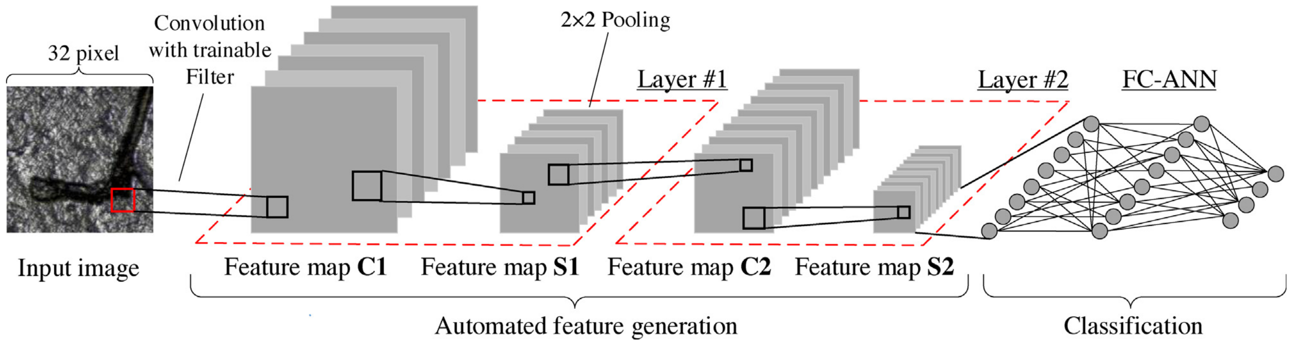

Fig. 1. Principle architecture of a deep CNN. Each layer is based on three processing steps, namely: convolution, non-linear activation (not shown) and feature pooling for dimension reduction.

Table 1
CNN configuration. The complexity of the network raises from *Basis* to *Depth-3*, additional layers in each schema are marked in bold.

Basis	*Depth-1*	*Depth-2*	*Depth-3*
Layer-1			
conv3-NK$_1$	conv3-NK$_1$	conv3-NK$_1$	conv3-NK$_1$
	conv3-NK$_1$	conv3-NK$_1$ **pad**	conv3-NK$_1$ pad
Max-pooling	Map-pooling	Map-pooling	Map-pooling
Layer-2			
conv3-	conv3-	conv3-NK$_2$	conv3-NK$_2$
	conv3-NK$_2$	conv3-NK$_2$ **pad**	conv3-NK$_2$ pad
		conv3-NK$_2$	conv3-NK$_2$ **pad**
Max-pooling	Max-pooling	Max-pooling	Max-pooling
Layer-3			
conv3-	conv3-	conv3-	conv3-NK$_3$ **pad**
		conv3-	conv3-NK$_3$ **pad**
			conv3-NK$_3$
			conv3-NK$_3$
			Max-pooling
FC-1024			
FC-1024			
FC-12			
Softmax			

stages (*Layer* – 1...3) of feature extraction, and a final classifier using a standard fully connected (FC) ANN with 2 hidden layer and 1024 neurons each. The 12-dimensional output via softmax regression represents the 12 different categories (6 defective and 6 non-defective) of the detection problem and the results are discussed in Section 4. The presented network design heuristics are strongly inspired by [14,15] due to its simplicity and effectiveness.

From *Basis* to *Depth-3* configuration, we add additional convolution (conv) layers to investigate the influence of depth (number of layers) of the CNN towards the detection result. This leads to a <*conv3-NK$_X$*> configuration, while *conv3* represents a fixed filter size of 3×3 pixel for each neuron in the network, followed by the number of neurons in each layer *NK$_X$*. The number *NK$_X$* in Table 2 represents the width of each layer and therefore the number of filters (neurons). Here we investigate the influence of adding neurons per layer towards the detection result.

The term *pad* refers to additional padding of the filters in the border region of the image. After each convolution layer, a non-linear ReLU activation is applied but not visualized for the sake of clarity. Spatial pooling is performed with max operation on non-overlapping 2×2 pixel neighbourhood following some of the convolutional layers. Each bold entry refers to a new non-linear neuron-layer with respect to the previous configuration.

Table 2
Configuration of the number of filters (neurons) for each layer.

	Number of filter (neurons)		
	NK$_1$	NK$_2$	NK$_3$
Basis	6	16	120
Width-1	16	32	256
Width-2	32	64	512
Width-3	64	128	256

4. Results

The following section presents the results of the hyper-parameter analysis for deep CNN for industrial inspection. All different defect categories are represented in one single CNN architecture.

4.1. Data and hyper-parameter

In this work, we evaluate the deep CNN architecture with a dataset of weakly supervised learning examples for industrial optical inspection [16]. Fig. 2 shows examples from the dataset and detection results respectively. An ellipse coarsely labels the defect region. The dataset consists of six classes, each containing 1000 defect free images and 150 defective of size 512×512 pixel. From here we extract 324,800 blocks of size 32×32 pixel. To avoid overfitting a data augmentation process is performed on the extracted blocks using linear transformations, i.e. rotations and mirroring. The complete set of images now contains 1,299,200 examples. The examples are shuffled randomly and split into training (70%), testing (15%), and the remaining for evaluation. Training of CNN requires roughly 24 h.

The main focus of the investigation is the influence of width (number of neurons/layer) and depth (number of layers) towards the detection result. Additional hyper-parameter have to be defined and will be explained in the following. The learning rate for SGD starts with a value of 4×10^{-3} and halves each 175,000 training examples. The batch size of SGD in training is 128 examples, the initialization of the L2 regularization $\lambda = 1 \times 10^{-5}$ which results in a total training time of 24 h.

4.2. Detection performance

The first investigation focuses the implication of the network depth towards the detection performance, by using the same width (number of neurons) on different depth (number of layers) configurations. *Base* width configuration with number of kernels *NK* = {6, 16, 120} are used throughout the network depth evaluation. The results are shown in Table 3.

Table 3
Detection and runtime results of the CNN with different depth configurations per 32×32 pixel block in the forward pass.

Depth	Max. accuracy (%)	# of param. (million)	Feature dimension	Time (ms)
Basis	94.68	6.98	NK$_3 \times 12 \times 4$	0.83
Depth-1	96.44	5.14	NK$_3 \times 11 \times 3$	0.78
Depth-2	96.87	2.32	NK$_3 \times 9 \times 1$	0.69
Depth-3	97.03	3.98	NK$_3 \times 7 \times 3$	1.01

The results demonstrate that adding layers of feature extraction boosts the detection result. From *Basis* configuration to *Depth-3* configuration, the accuracy of defect detection rises around 2.35%. The *Basis* configuration is represented by the biggest amount of parameters but more efficient compared to *Depth-3* configuration. This effect is related to more convolution stages in the *Depth-3* case. Here, more convolution stages require more runtime to be propagated through the network. At the same time the feature dimension is smaller compared to *Basis* configuration, as Max-pooling reduces the dimensionality. For the investigation of the effect of adding width to the convolution layer, we will focus on *Depth-2* and *Depth-3* configuration, as they represent a trade-off between runtime and accuracy.

The analysis of adding filters (neurons) to the convolution layers is represented in Table 4. From layer to layer, the number of filters *NK$_X$* rises with respect to the configurations in Tables 1 and 2.

The analysis shows, comparable to the previous investigation, that adding neurons in each layer contributes towards better detection performance. From *Basis* to *Width-3* setup, the accuracy rises by 1.32% in *Depth-2* case and 1.34% in *Depth-3* case. Nevertheless, the impact of adding filters is comparatively low. An explanation of this low performance gain can be the limited

Table 4
Detection results of the CNN with different width configurations per 32×32 pixel block in the forward pass.

	Best accuracy @ time in (ms)		
Width	NK	*Depth-2*	*Depth-3*
Basis	{6, 16, 120}	95.87% @ 0.69	96.03% @ 1.01
Width-1	{16, 32, 256}	96.94% @ 0.98	97.19% @ 1.60
Width-2	{32, 64, 512}	97.13% @ 1.65	97.36% @ 3.25
Width-3	{64, 128, 256}	97.19% @ 1.42	97.37% @ 2.00

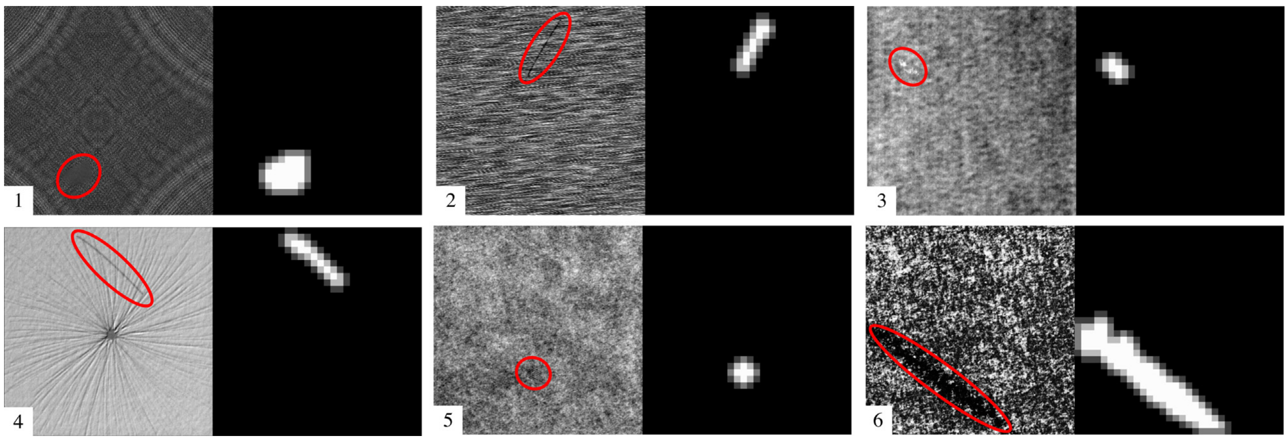

Fig. 2. Original image and detection result of the CNN. One network is trained to detect all different defect categories, applied to 512 × 512 pixel images, where the number in the images indicates one out of six defect categories.

geometrical complexity of defects. A network processes raw data in stages towards complex representations. In case of a defect, the complexity is limited, so that adding more neurons will lead to a saturation effect and the CNN is not able to generate more complex features.

For the final CNN for industrial inspection based on the presented data we suggest the *Depth-2/Width-3* setting as it shows the best trade-off between both, accuracy and runtime.

The overall detection performance is represented in Table 5. The table shows the true positive rate (TPR) of defect free examples classified as defect free and the true negative rate of defects classified in the correct category in comparison with existing approaches. The deep CNN results represent the work in this contribution. Except the TPR in class 3 and slightly the TNR in class 6, the deep CNN outperforms all existing techniques with respect to the overall detection accuracy, although no prior knowledge or manual input was feed into the CNN.

Table 5
Detection results of the CNN with respect to existing techniques for complete 512 × 512 pixel image analysis.

Class	Deep CNN	Statistical features [5]	SIFT and ANN [17]	Weibull [18]
TPR (%)				
1	100	99.4	98.9	87.0
2	100	94.3	95.7	–
3	95.5	99.5	98.5	99.8
4	100	92.5	–	–
5	98.8	96.9	98.2	97.2
6	100	100	99.8	94.9
TNR (%)				
1	100	99.7	100	98.0
2	97.3	80.0	91.3	–
3	100	100	100	100
4	98.7	96.1	–	–
5	100	96.1	100	100
6	99.5	96.1	100	100
Average accuracy (%)				
	99.2	95.9	98.2	97.1

5. Conclusion

In this work an approach in visual defect detection using deep machine learning, namely deep CNN, is presented. The performance of the proposed approach is measured on a data set representing 12 different classification categories with visual defects occurring on heavily textured background. As opposed to hand-crafting features on pixel level, with CNN we engineer architectures by investigating different hyper-parameters involved in the process. In this way, systems for OQC can be developed with minimum prior knowledge on the problem domain.

To extract meaningful features from raw data, a deep architecture requires huge amounts of training data. In case the generation of training data is expensive, deep learning might not be an appropriate technology. Nevertheless, we believe, that this data driven approach of deep learning algorithms will be a key technique in big data analysis in different application domains in manufacturing.

Acknowledgement

The authors gratefully acknowledge the financial support by DFG (German Research Foundation) for Subproject B5 Reliable Processes within the CRC 747 Micro Cold Forming – Processes, Optimization, Characterization. Daniel Weimer was funded by German Academic Exchange Service (DAAD) ID-57044996.

References

[1] Xie X (2008) A Review of Recent Advances in Surface Defect Detection Using Texture Analysis Techniques. *Electronic Letters on Computer Vision and Image Analysis* 7(3):1–22.
[2] Neogi N, Mohanta DK, Dutta PK (2014) Review of Vision-Based Steel Surface Inspection Systems. *EURASIP Journal on Image and Video Processing* 1–19.
[3] Pernkopf F, O'Leary P (2002) Visual Inspection of Machined Metallic High-Precision Surfaces. *EURASIP Journal on Applied Signal Processing* 7:667–668.
[4] Jiang X, Scott P, Whitehouse D (2008) Wavelets and Their Applications for Surface Metrology. *CIRP Annals – Manufacturing Technology* 57(1):555–558.
[5] Scholz-Reiter B, Weimer D, Thamer H (2012) Automated Surface Inspection of Cold-Formed Micro Part. *CIRP Annals – Manufacturing Technology* 61(1):531–534.
[6] Hu SJ, Ko J, Weyand L, ElMaraghy HA, Lien TK, Koren Y, Bley H, Chryssolouris G, Nasr N, Shpitalni M (2011) Assembly System Design and Operations for Product Variety. *CIRP Annals – Manufacturing Technology* 60(1):715–733.
[7] Scholz-Reiter B, Hamann T (2008) The Behaviour of Learning Production Control. *CIRP Annals – Manufacturing Technology* 57(1):459–462.
[8] Hubel DH, Wiesel TN (1962) Receptive Fields, Binocular Interaction, and Functional Architecture in the Cat's Visual Cortex. *Journal of Physiology* 160:106–154.
[9] LeCun Y, Buttou L, Bengio Y, Haffner P (1998) Gradient-Based Learning Applied to Document Recognition. *Proceedings of the IEEE* 86(11):2278–2324.
[10] LeCun Y, Bengio Y, Hinton GE (2015) Deep Learning. *Nature* 521:436–444.
[11] Nair V, Hinton GE (2010) Rectified Linear Units Improve Restricted Boltzmann Machines. *Proceedings of the 27th International Conference on Machine Learning*, 807–814.
[12] Hinton GE, Salakhutdinov R (2006) Reducing the Dimensionality of Data with Neural Networks. *Science* 313:504–507.
[13] Srivastava N, Hinton GE, Krizhevsky A, Sutskever I, Salakhutdinov R (2014) Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *Journal of Machine Learning Research* 15:1929–1958.
[14] Simonyan K, Zisserman A (2014) *"Very Deep Convolutional Networks for Large-Scale Image Recognition", CoRR abs/1409.1556.*
[15] Masci J, Meier U, Ciresan D, Schmidhuber J, Fricout G (2012) Steel Defect Classification with Max-Pooling Convolutional Neural Networks. *Proceedings of International Joint Conference on Neural Networks*, 1–6.
[16] Weakly Supervised Learning for Industrial Optical Inspection. *29th Annual Symposium of the German Association for Pattern Recognition.*
[17] Siebel NT, Sommer G (2008) Learning Defect Classifiers for Visual Inspection Images by Neuro-Evolution Using Weakly Labelled Training Data. *Evolutionary Computation* 3925–3931.
[18] Timm F, Barth E (2011) Non-Parametric Texture Defect Detection Using Weibull Features, IS&T/SPIE Electronic Imaging. *International Society for Optics and Photonics.*