Anomaly detection in formed sheet metals using convolutional autoencoders

Heger, Jens; Desai, Gururaj; Zein El Abdine, Mazhar

Published in:
Procedia CIRP

DOI:
10.1016/j.procir.2020.04.106

Publication date:
2020

Document Version
Publisher's PDF, also known as Version of record

Available online at www.sciencedirect.com

ScienceDirect

Procedia CIRP 93 (2020) 1281–1285

www.elsevier.com/locate/procedia

53rd CIRP Conference on Manufacturing Systems

Anomaly detection in formed sheet metals using convolutional autoencoders

Jens Heger, Gururaj Desai, Mazhar Zein El Abdine*

Institute of Product and Process Innovation, Leuphana Universität Lüneburg, Universitätsallee 1, 21335 Lüneburg, Germany

* Corresponding author. *E-mail address:* mazhar.abdine@leuphana.de

Abstract

Autoencoders are used to compress data and learn how to reliably reconstruct it. Through comparing the reconstruction error with a set threshold, they are able to detect anomalies in unseen datasets where the data does not quite match the reconstructed input samples. In this work, we attempt to investigate the use of convolutional autoencoders in the field of visual quality inspection, where images of formed sheet metals from a real production line are inspected for the occurrence of cracks and wrinkle formation. This approach tackles the problem of needing enough defective samples to attain reliable detection accuracies.

Keywords: Surface quality inspection, process monitoring, artifical intelligence, deep autoencoders

1. Introduction

After being formed into the desired shape, sheet metals in the automotive industry undergo a manual quality control check at the end of the production line by highly trained personnel. These personnel look for production defects, such as wrinkles and cracks and sort them out, preventing defective products from undergoing further finishing and assembly. Though manual inspection has been employed for a long time in the industry, manufacturers have recently turned to machine vision solutions to automate the quality control phase, thereby reducing costs and increasing the speed and efficiency of the process. Audi AG, for example, have begun testing the integration of deep learning into smart cameras installed inside the press shops for series production. They spent several months training artificial neural networks with several million labeled images. The biggest challenge, as reported [1], was the creation of such a large database of images with the proper labels. Furthermore, the system has to be reconfigured for every different component to be produced, such as doors, engine hoods, fenders etc. The former sheds light on a general challenge in regards to training deep neural networks for automatic surface inspection: well-optimized manufacturing processes produce mostly non-defective samples (majority class), and the amount of defective samples (minority class) is not large enough for the algorithms to perform up to industrial requirements. The ratio of non-defective to defective products is often highly imbalanced, ranging from 9:1 up to 1000000:1 [2]. In this paper, we propose the use of a different type of deep neural networks, namely convolutional autoencoders (CAE), in order to address the challenge of needing sufficiently large amounts of training data, specifically from the minority class. In contrast to the classical convolutional neural network (CNN) based approaches, CAE is a *self-supervised* method which only requires unlabeled training samples from the majority class. Thus, the problem statement shifts from classification to anomaly detection. We trained and tested a CAE network on a

10.1016/j.procir.2020.04.106

set of images obtained from a real production line and compared its performance with a state of the art CNN.

1.1. State of the art

After being proposed by LeCun et al. [3], CNNs along with their variants have been proven to be the most effective methods in image classification and object detection. The variants consist of networks with slightly different structures having the goal of extracting more complex non-linear features and improving the generalization ability of the model. CNNs generally scale very well with more data. In the field of surface quality inspection, these algorithms proved to be highly accurate in detecting and classifying defects obtained from public databases [4, 5]. Moreover, when coming across small datasets acquired from a real production plant, the performance of the classical CNN is not up to par. In that case, researchers have used transfer learning approaches [6, 7] to optimize performance, i.e. learning features and pre-trained networks from different domains and using them for their own models. However, the application of transfer learning in steel surface inspection is not as good as in other fields, as is explained by Di et al. in [8]. Image context and features of steel surfaces are different from most pre-trained models, which violates the application conditions of transfer learning.

Besides transfer learning, several other methods have been proposed to deal with class imbalances, such as data augmentation [9], where additional synthetic samples of the minority class are created through scaling the original image, application of different filters, etc. Undersampling of the majority class through reducing the amount of input samples can be also viable, but would potentially leave out important features of the data. A detailed list of these methods which include readjusting the training data distribution or modifying the learning procedure itself can be found in [10, 11]. These methods, however, have some drawbacks, as they can generate unstable results.

1.2. Autoencoders

Autoencoders (AE) are a type of neural network that are trained to efficiently compress and encode their input and learn how to reliably reconstruct it based on the reduced encoded representation.

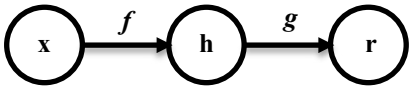

Fig. 1. The general structure of an AE.

The standard AE can be viewed as a network with three parts [12], namely:

- An encoder network that maps x to h.
- The output of the encoder, which is a low-dimensional latent representation of x. It prevents the AE from learning a trivial identity function.
- A decoder network that maps h to r.

The encoder is trained to minimize the reconstruction loss $L = (x, r)$ which can be any error metric such as the Euclidean distance or the mean squared error (MSE). For any new input y, the reconstruction loss $L = (y, g(f(y)))$ is compared to the original threshold T obtained after training. If it exceeds it, it is sorted as an anomaly. In this work, T is given as the sum of the mean and the standard deviation of the reconstruction losses calculated after the training phase. This threshold is set based on our prior observations and experiments on several products, and as such does not undergo any optimization in this work.

AEs first gained traction through the work of Hinton and Salakhutdinov [13] for dimensionality reduction purposes. In their paper, they trained a deep AE with gradually smaller hidden layers, ending in a bottleneck of 30 units. Their model yielded less reconstruction error than the classical principal component analysis dimensionality reduction algorithm into 30 dimensions, and the learned representation was qualitatively easier to interpret. AEs constitute a special form of supervised learning techniques, namely self-supervised learning, where the targets are generated from the input data. Other than dimensionality reduction, they are used for various applications nowadays including image denoising, image coloring, data compression, speech and text recognition and generation. Recent works [14, 15] have shown that the use of AEs for the feature extraction phase of CNNs can improve performance, which has been also successfully tested for certain real-world scenarios for metal surfaces [16, 17]. The main difference between the classical CNNs and CAEs is that the former are trained to learn convolutional filters and combine features with the aim of classifying the inputs into the targeted classes, while the latter are trained only to learn filters that would extract features able to reconstruct the input. The standard AE, however, is not suitable for processing images as it ignores the 2D image structure. Hence, we used a CAE instead. In a CAE, the dense layers of a standard AE are replaced with convolutional- and subsequently pooling layers.

2. Approach

2.1. Dataset

The dataset used in this work was acquired from a camera installed inside a pressline after a certain stage of pressing, and not at the end of the production line. It consists of 13,000 images, 2,600 of which are defective (25 %). We deliberately chose the product that is exhibiting a high number of defects in order to examine how the algorithms behaves with a varying minority class training data size.

2.2. Pre-processing

The images are originally of one channel with the size of 2592 × 1944 pixels. Before using them for training, a relevant region of interest (ROI) is defined by an expert, indicating where the defects are most likely to appear. The images undergo further scaling to 256 × 256 pixels, and two filters are

applied on them to better detect edges. Fig. 2 shows an example of the defects seen, along with the surface appearance before and after applying the filters.

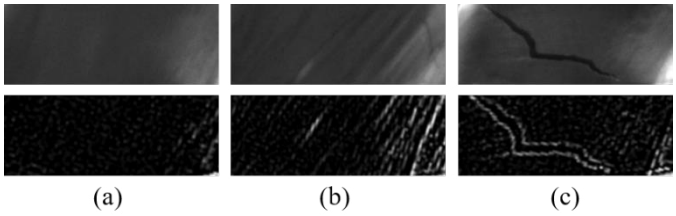

Fig. 2. (a) Non-defective surface; (b) Defective wrinkled surface; (c) Defective cracked surface.

2.3. Network Architecture

The specific parameters for the encoder and the decoder networks are shown in Tables 1 and 2, respectively. The input images are of size 256 × 256 pixels and undergo further processing by three convolutional layers in the encoder as well as in the decoder, with the rectified linear unit (ReLU) as the non-linear activation function. The max-pooling and up-sampling layers shown in orange and green in Fig. 3 work opposite to each other; the former reduces the dimensions of the feature maps, forcing the encoder to learn more sparse features while the latter expands the dimensions of the feature maps back to the original input size.

Table 1. Encoder network parameters.

Layer type	Filters	Size/stride	Output shape
Input			(256,256,1)
Convolutional	64	3×3/1	(256,256,64)
Max-pooling		2×2/2	(128,128,64)
Convolutional	32	3×3/1	(128,128,32)
Max-pooling		1×1/1	(128,128,32)
Convolutional	16	3×3/1	(128,128,16)
Max-pooling		2×2/2	(64,64,16)

Table 2. Decoder network parameters.

Layer type	Filters	Size/stride	Output shape
Convolutional	16	3×3/1	(64,64,16)
Up-sampling		2×2/2	(128,128,16)
Convolutional	16	3×3/1	(128,128,16)
Up-sampling		1×1/1	(128,128,16)
Convolutional	31	1×1/1	(128,128,32)
Up-sampling		2×2/2	(256,256,32)
Convolutional	1	3×3/1	(256,256,1)

As for the CNN model used, our architecture is similar to the state of the art CNN proposed by Chollet in [18] to build powerful image classifiers using only a few thousand training examples. It consists of three convolutional and max-pooling pairs as the feature extractors and two dense layers as the classifiers. The model also includes batch normalization and two dropout layers to help against overfitting. The hyper-parameters, such as the activation functions, number of filters, learning rate and number of hidden layers were tuned using the 'Talos' library [19], which combines all possible parameters in a grid and chooses the best combinations to minimize the loss or maximize evaluation accuracy.

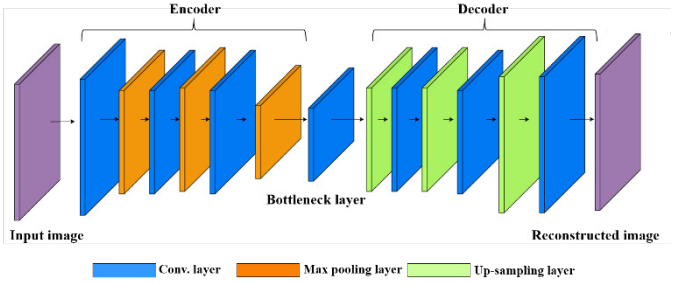

Fig. 3. The network architecture of the CAE used in this work.

3. Results

3.1. Training and testing

We trained both models using various subsets of the full data. For the CNN model, we used 10,000 non-defective samples and varied the number of defective samples used for training as follows: 200, 400, 600, 800, 1000, 1500, 2000, and 2500. The combined samples were then split into 80% for training and 20 % for testing. As for the CAE model, we only used non-defective samples for training in the order of 500, 1000, 1500, 2000, 4000, 6000, 8000, 10000. The samples were split into 90 % for training and 10 % for testing. The test set also included additional 2,500 defective images to test how well the algorithm can detect these defects. To counteract the stochastic nature of the algorithms, both models were trained and evaluated 10 times for 30 epochs each. The results for each subset are then the average of the 10 trials. The models were trained on Keras 2.3.1 using Tensorflow 1.14.0 as the backend.

3.2. Evaluation

For evaluating algorithm performance, we chose one of the standard classification metrics, namely the fraud recall metric. It is defined as $R = TP/(TP + FN)$, where TP denotes the number of defective samples that were correctly classified as defective, while FN denotes the number of defective samples that were falsely classified as non-defective. In other words, fraud recall represents the ratio of correct positive (defective) predictions to the total number of defective samples. A high recall rate corresponds to a low FN rate. In the automotive industry, high FN rates are much more costly than a false alarm (FP). As such, we have chosen to put heavy emphasis on this metric in our evaluation. Furthermore, the general accuracy shown in Fig. 4 and Fig. 5 depicts the ratio of correct positive and negative predictions to the total number of samples in the dataset.

3.3. CNN

Fig. 4 and Table 3 show the results for the benchmark CNN algorithm. It can be seen that while the general accuracy reached was above 95 % for all subsets, the fraud recall was only 49.5% when training with 10,000 non-defective samples and 800 defective samples. It later peaks at 92.8 % when training with 10,000 non-defective samples and 2,000 defective samples. Within the range of 200 to 1,000 defective samples used for training, i.e. when the ratio of defective to non-defective samples is between 2 % and 10 %, the algorithm performed poorly, as the fraud recall was under 65.7 %. The general accuracy is consistently high due to the large number of non-defective samples included in the training and test sets.

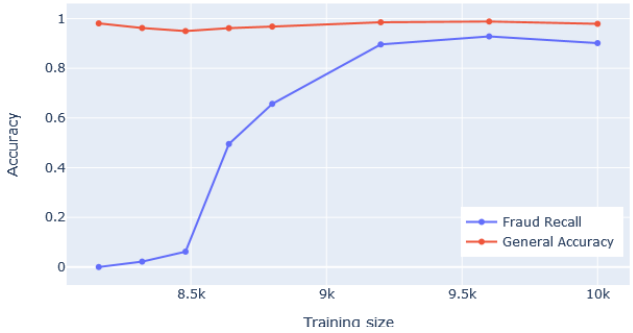

Fig. 4. CNN model performance in classifying defects while trained using various subsets of the full data.

Table 3. The detailed dataset sizes before being split for training and testing along with the results for the CNN model

Dataset size	Defective samples included	Ratio defective to non-defective	General accuracy (%)	Fraud recall (%)
10,200	200	0.02	98	0
10,400	400	0.04	96.2	2.1
10,600	600	0.06	95	6.2
10,800	800	0.08	96.2	49.5
11,000	1,000	0.1	96.8	65.7
11,500	1,500	0.15	98.5	89.6
12,000	2,000	0.2	98.8	92.8
12,500	2,500	0.25	97.9	90.1

3.4. CAE

The CAE algorithm achieved a maximum detection accuracy of 96.5 % when trained with a dataset of 10,000 non-defective samples. The algorithm converges slightly slower than CNN, however the fraud recall continuously improves from 78.9 % for 2,000 training samples to 82.3 % for 4,000 samples and ultimately 95.5% for the full dataset. The calculated threshold T used in order to classify the test set is presented in Fig. 6, where it is clear that a large majority of reconstruction errors exhibited by the defects exceeds this set threshold. In this case, this amounts to 95.5 % of all defective samples, specifically 2,387 samples.

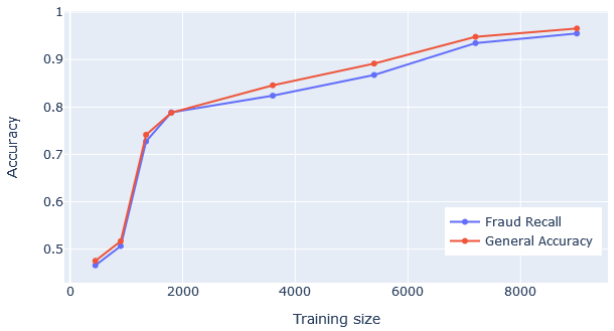

Fig. 5. CAE model performance in detecting anomalies/defects while trained using various subsets of the full data.

Table 4. The detailed dataset sizes along with the results for the CAE model

Dataset size	Defective samples included	General accuracy (%)	Fraud recall (%)
3,000	2,500	47.5	46.5
3,500	2,500	51.6	50.6
4,000	2,500	74.1	72.7
4,500	2,500	78.8	78.8
6,500	2,500	84.5	82.3
8,500	2,500	89.1	86.7
10,500	2,500	94.8	93.4
12,500	2,500	96.5	95.5

Fig. 6. Histogram of the reconstruction errors along with the threshold calculated by the CAE model.

4. Discussion

In this work, we tested a CAE on a dataset of non-defective and defective images obtained from a real production line and compared its performance with a state of the art CNN. While the CNN was able to correctly classify non-defective images as being non-defective with a very high accuracy, it failed to do so for unseen defective images. The algorithm required at least 1,500 defective samples to be used, which corresponds to 13 % of the total dataset size, in order to achieve reliable fraud recall rates. In cases where the production line does not produce many defects, a standard CNN would most likely fail in correctly classifying defective samples as being defective. On the other

hand, a CAE algorithm is able to extract important features solely from non-defective images and through the use of the reconstruction errors, it is able to reliably detect unseen anomalous samples, regardless of where the anomalies appear within the ROI. In future work, we will look to dynamically optimize the classification threshold of the CAE algorithm in order to find the optimal decision boundary for every separate region defined. Through splitting the image into multiple patches we aim to isolate relevant regions of the image and disregard the noise in it. By doing so, an expert to manually define a relevant ROI would no longer be required.

Acknowledgements

The authors gratefully acknowledge the financial support given by the Federal Ministry for Economic Affairs and Energy (BMWi) under the Central Innovation Programme for SMEs (ZIM) for the project “SmartPress” on the basis of a decision of the German Bundestag (grant number: ZF4084503GR7).

References

[1] Durian M, 2018. Audi testing artifical intelligence to improve quality control and production. Available at: https://www.c-magazine.com/news/audi-testing-artificial-intelligence-to-improve-quality-control-and-production. [Accessed 28 January 2020].

[2] Wang J, Ma Y, Zhang L, Gao RX, Wu D, 2018. Deep learning for smart manufacturing: Methods and applications. J Manuf Syst. doi:10.1016/j.jmsy.2018.01.003.

[3] LeCun Y, Bengio Y, Hinton G, 2015. Deep learning. Nature 521, pp. 436–444. doi: 10.1038/nature14539.

[4] Staar B, Lütjen M, Freitag M, 2018. Anomaly Detection with Convolutional Neural Networks for Industrial Surface Inspection, in: 12th CIRP Conference on Intelligent Computation in Manufacturing Engineering - CIRP ICME 2018. Elsevier B.V, Amsterdam, NL, 2018, pp. 484-489.

[5] Natarajan V, Hung TY, Vaikundam S, Chia LT, 2017. Convolutional networks for voting-based anomaly classification in metal surface inspection, in: IEEE International Conference on Industrial Technology (ICIT): pp. 986-991.

[6] Seunghyeon K, Wooyoung K, Yung-Kyun N, Park FC, 2017. Transfer learning for automated optical inspection, in: International Joint Conference on Neural Networks (IJCNN), Anchorage, AK, pp. 2517-2524. doi:10.1109/IJCNN.2017.7966162.

[7] Oquab M, Bottou L, Laptev I, Sivic J, 2014. Learning and Transferring Mid-level Image Representations Using Convolutional Neural Networks, in: 2014 IEEE Conference on Computer Vision and Pattern Recognition. doi: 10.1109/cvpr.2014.222.

[8] Di H, Ke X, Peng Z, Dongdong Z, 2019. Surface defect classification of steels with a new semi-supervised learning method. Optics and Lasers in Engineering, 117, 40–48. doi:10.1016/j.optlaseng.2019.01.011.

[9] Arikan S, Varanasi K, Stricker D, 2019. Surface Defect Classification in Real-Time Using Convolutional Neural Networks. ArXiv abs/1904.04671: n. pag.

[10] Haibo He, Garcia, EA, 2009. Learning from Imbalanced Data. IEEE Transactions on Knowledge and Data Engineering, 21(9), 1263–1284. doi:10.1109/tkde.2008.239.

[11] Khan, SH, Hayat M, Bennamoun M, Sohel FA, Togneri R, 2018. Cost-Sensitive Learning of Deep Feature Representations From Imbalanced Data, in: IEEE Transactions on Neural Networks and Learning Systems, 29(8), 3573–3587. doi:10.1109/tnnls.2017.2732482

[12] Goodfellow I, Bengio Y, Courville A, 2016. Deep Learning. MIT Press, Cambridge, MA, USA, p. 499 – 523.

[13] Hinton GE, Salakhutdinov RR, 2006. Reducing the Dimensionality of Data with Neural Networks. Science 313, 504-507. doi: 10.1126/science.1127647.

[14] Cavallari GB, Ribeiro LS, Ponti MA, 2018. Unsupervised Representation Learning Using Convolutional and Stacked Auto-Encoders: A Domain and Cross-Domain Feature Space Analysis, in: 31st SIBGRAPI Conference on Graphics, Patterns and Images (SIBGRAPI): pp. 440-446.

[15] Masci J, Meier U, Cireşan D, Schmidhuber J, 2011. Stacked convolutional auto-encoders for hierarchical feature extraction, in: Honkela, T., Duch, W., Girolami, M., Kaski, S. (eds.) ICANN 2011. LNCS, vol. 6791, pp. 52–59. Springer, Heidelberg. doi: 10.1007/978-3-642-21735-7_7.

[16] Essid O, Laga H, Samir C, 2018. Automatic detection and classification of manufacturing defects in metal boxes using deepneural networks, in: PLoS ONE 13(11): e0203192. doi: 10.1371/journal.pone.0203192

[17] Tao X, Zhang D, Ma W, Liu Xong, Xu D, 2018. Automatic Metallic Surface Defect Detection and Recognition with Convolutional Neural Networks, in: Applied Sciences. 8. 1575. doi: 10.3390/app8091575.

[18] Chollet F. Building powerful image classification models using very little data. Available at: https://blog.keras.io/building-powerful-image-classification-models-using-very-little-data.html [Accessed 28 Janurary 2020].

[19] Autonomio Talos [Computer Software], 2019. Available at: http://github.com/autonomio/talos. [Accessed 06 January 2020].