Automating Industrial Visual Inspection Systems with Deep Reconstruction-based Anomaly Detection

Pietro Buzzega[1], **Cynthia I. Ugwu**[2], **Alberto Cenzato**[1], **Massimo Regoli**[1], **Franz Tschimben**[1], **Marco Boschetti**[1], **Oswald Lanz**[1,2]

[1] Covision Lab, Bressanone-Brixen, Italy
[2] Free University of Bozen-Bolzano
name.surname@covisionlab.com
https://www.covisionlab.com

Abstract

The increasing longing to optimize manufacturing procedures led us throughout industrial revolutions, increasing consumers' expectations and consequently creating the suppliers' need to fulfill them. Quality control represents a crucial part of automatic production processes nowadays; the better the quality, the less need for human intervention. In this work, we present our approach to automate visual inspection systems: leveraging state-of-the-art reconstruction-based anomaly detection approaches, we aim at finding defects in an unsupervised manner, avoiding time-consuming manual or automatic labeling techniques.

1 Introduction

Human labeling still plays a fundamental role in the industry. Despite automatic vision systems having been at disposal for more than 30 years, it is not uncommon to see operators discriminating between good and defective parts right next to the production line. These human experts are valuable to the company and sometimes unique: since the market demand decides the number of manufactured pieces, managing work shifts is not always straightforward. Moreover, even when setting up automated systems becomes compelling, the latter's need for labeled data makes human knowledge essential.

While labeling is expensive and time-consuming, acquiring data is not a big deal. On the one hand, most companies already have a vision system capable of capturing objects passing through; on the other hand, configuring one just for taking pictures does not represent a challenge. Given the vast amount of unlabeled data available, unsupervised learning appears to be the easiest way out.

Fortunately, no self-respecting production line would yield a considerable amount of scrap; therefore, we can assume that most of the images depict OK pieces. If we can fit that unlabeled set of data, we may be sure that we learned the non-defective distribution: any part that does not conform would be discarded. This concept perfectly blends with the task of anomaly detection: if we can find anomalies, we may find defects.

2 Anomaly detection

Anomaly detection, also called outliers or novelty detection, aims at identifying anomalous patterns that are different from those seen in regular instances. For example, in computer vision applications such as defect detection, an anomaly is any image or image portion which exhibits significant variation from the predefined characteristics of normality [Mishra *et al.*, 2021]. Other critical machine learning applications rely on anomaly detection: fraudulent financial transactions, medical image analysis, and video surveillance, to name a few. For such tasks, it is difficult and expensive to collect abnormal data [Kimura and Yanagihara, 2019], leading the majority of research efforts to focus on unsupervised and self-supervised anomaly detection that employ normal data only.

2.1 Related work

One-Class Classification (OCC) is broadly used for unsupervised anomaly detection. It branches out in kernel-based methods like One-Class SVM (OC-SVM) [Schölkopf *et al.*, 1999] and boundary methods such as Support Vector Data Description (SVDD) [Tax and Duin, 2004]. However, these fail with high-dimensional data due to the curse of dimensionality [Ruff *et al.*, 2018], while deep-learning-based methods like [Oza and Patel, 2019; Ruff *et al.*, 2018] exploit that high-dimensionality by capturing more complex features: the latter guarantees them a substantial performance gap.

Sticking to the unsupervised paradigm, reconstruction-based models such as autoencoders [Rumelhart *et al.*, 1985], generative adversarial networks [Perera *et al.*, 2019], or transformers [Mishra *et al.*, 2021] can be trained to reconstruct or generate normal data, also modeling the learned latent space using a target distribution.

Moving on to self-supervised learning, instead, deep neural networks can be used to solve pretext tasks like predicting geometric transformation, performing contrastive learning [Sohn *et al.*, 2021] or clustering [Caron *et al.*, 2018]. This way, the network learns high-level semantic features, useful for detecting anomalies.

Due to its simplicity and reliable performance, we adopt a reconstruction-based approach. Specifically, we reconstruct images with a convolutional *Denoising AutoEncoder* (DAE) [Vincent *et al.*, 2008].

3 System architecture and workflow

Acquiring images serves as the first step of our workflow. A proper acquisition system would be repeatable, producing pictures whose basic statistics do not change over time or place. As mentioned above, most industrial plants already dispose of a reliable acquisition system: if this were not the case, its implementation would not be a technical or scientific challenge; we thus give that for granted in this work.

3.1 Preprocessing

Subsequently, we should ask ourselves whether the examined part is rigid or can change shape. The answer would determine whether a registration[1] phase could be advantageous. When the object at issue does not exhibit any change in its silhouette, aligning pictures typically lowers the pixel-wise variability of the dataset exceptionally. Also, it allows to easily segment out the background, as pieces always hold the same position. The two latter points heavily simplify the learning stage: a shallower network on a registered dataset could perform as a deeper network on a non-registered one; at times, this is fundamental for the learning to happen. Our use-cases usually provide for the production of metal parts; hence, we rely on Generalized Hough Transform [Hough, 1962] to align images.

3.2 Model

The core of our architecture consists of a denoising autoencoder network. To generalize fast to new images, which could vary in quality and resolution, we designed a network architecture based on the ResidualBlock [He *et al.*, 2016] that adapts automatically to different input shapes. The training phase of a DAE involves adding a certain amount of noise (Gaussian noise, in our case) to each image and feeding the result to the network; the latter yields another image with the same shape of the input. We optimize our objective – minimizing the *Mean Squared Error* between input and output – via a gradient descent optimizer. After proper training, the network can reconstruct the input almost correctly. However, since the information needs to flow throughout the bottleneck, which is much smaller than the image, less frequent features are left out; as a result, anomalies cannot be reconstructed. The higher the pixel-wise difference between input and output for a given patch, the likelier that patch is anomalous.

3.3 Classification

The difference between input and output (*i.e.* heatmap) links each pixel with an anomaly score. The most common technique to provide a score for each image is summing all the anomaly scores for each pixel. However, in production environments, this does not suffice. We eventually need a binary decision: any piece that exceeds a threshold (usually chosen as the best for the validation set) is considered anomalous.

While this simple rule works if all anomalies are also defects, this is not always the case. For instance, we could mistake light change, noise, or dirt for defects; for this reason, we might need a final classification phase that takes heatmaps as input and outputs binary decisions. Assuming we could have access to a small amount of labeled data, we make use of either a pre-trained network or Hu moment invariants [Hu, 1962] to extract meaningful features from the heatmaps. Then, we exploit the latter to train a Multilayer Perceptron in a supervised manner and use it as the final classifier.

[1] Alignment of different pictures depicting the same piece.

4 Deployment

While we write Python programs for everything linked to neural network and algorithm development, the adoption of C++ for our deployment application makes it more performing and reliable. Thanks to ONNX [ONNX,] and the related run-time environment powered by Nvidia, models from Pytorch [Paszke *et al.*, 2019] or TensorFlow [Abadi *et al.*, 2015] can be easily exported and stored. ONNX files act as the interface between research and deployment: the Nvidia TensorRT framework loads and optimizes the real-time performance of our deep models.

Monitoring production statistics continuously plays a valuable role in the quality control process: detecting problems as soon as they appear delivers an immeasurable value for the client. To this end, we perform anomaly detection on the metrics produced by our system and notify any deviation from the usual behavior.

5 Conclusions and future work

Field testing our system confirmed us its valuable performance and generalization capabilities. However, discriminating anomalies from defects is not always easy and still requires a small amount of supervision. In the future, we will explore different architectures and algorithms to reduce the error rate, simultaneously reducing human intervention as much as possible.

References

[Abadi *et al.*, 2015] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org.

[Caron *et al.*, 2018] Mathilde Caron, Piotr Bojanowski, Armand Joulin, and Matthijs Douze. Deep clustering for unsupervised learning of visual features. In Vittorio Ferrari, Martial Hebert, Cristian Sminchisescu, and Yair Weiss, editors, *Computer Vision – ECCV 2018*, pages 139–156, Cham, 2018. Springer International Publishing.

[He *et al.*, 2016] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[Hough, 1962] Paul VC Hough. Method and means for recognizing complex patterns, December 18 1962. US Patent 3,069,654.

[Hu, 1962] Ming-Kuei Hu. Visual pattern recognition by moment invariants. *IRE transactions on information theory*, 8(2):179–187, 1962.

[Kimura and Yanagihara, 2019] Masanari Kimura and Takashi Yanagihara. Anomaly detection using gans for visual inspection in noisy training data. In Gustavo Carneiro and Shaodi You, editors, *Computer Vision – ACCV 2018 Workshops*, pages 373–385, Cham, 2019. Springer International Publishing.

[Mishra *et al.*, 2021] Pankaj Mishra, Riccardo Verk, Daniele Fornasier, Claudio Piciarelli, and Gian Luca Foresti. Vt-adl: A vision transformer network for image anomaly detection and localization. *2021 IEEE 30th International Symposium on Industrial Electronics (ISIE)*, Jun 2021.

[ONNX,] ONNX. Onnx, the open neural network exchange (onnx) is an open-source artificial intelligence ecosystem. http://https://onnx.ai.

[Oza and Patel, 2019] Poojan Oza and Vishal M. Patel. One-class convolutional neural network. *IEEE Signal Processing Letters*, 26(2):277–281, Feb 2019.

[Paszke *et al.*, 2019] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019.

[Perera *et al.*, 2019] Pramuditha Perera, Ramesh Nallapati, and Bing Xiang. Ocgan: One-class novelty detection using gans with constrained latent representations, 2019.

[Ruff *et al.*, 2018] Lukas Ruff, Robert Vandermeulen, Nico Goernitz, Lucas Deecke, Shoaib Ahmed Siddiqui, Alexander Binder, Emmanuel Müller, and Marius Kloft. Deep one-class classification. In Jennifer Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 4393–4402. PMLR, 10–15 Jul 2018.

[Rumelhart *et al.*, 1985] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning internal representations by error propagation. Technical report, California Univ San Diego La Jolla Inst for Cognitive Science, 1985.

[Schölkopf *et al.*, 1999] Bernhard Schölkopf, Robert Williamson, Alex Smola, John Shawe-Taylor, and John Platt. Support vector method for novelty detection. In *Proceedings of the 12th International Conference on Neural Information Processing Systems*, NIPS'99, page 582–588, Cambridge, MA, USA, 1999. MIT Press.

[Sohn *et al.*, 2021] Kihyuk Sohn, Chun-Liang Li, Jinsung Yoon, Minho Jin, and Tomas Pfister. Learning and evaluating representations for deep one-class classification, 2021.

[Tax and Duin, 2004] David Tax and Robert Duin. Support vector data description. *Machine Learning*, 54:45–66, 01 2004.

[Vincent *et al.*, 2008] Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. Extracting and composing robust features with denoising autoencoders. In *Proceedings of the 25th international conference on Machine learning*, pages 1096–1103, 2008.