FABLE : Fabric Anomaly Detection Automation Process

Simon Thomine, Hichem Snoussi, Mahmoud Soua

FABLE : Fabric Anomaly Detection Automation Process

Simon Thomine
University of technology of Troyes
AQUILAE
Troyes, France
simon.thomine@utt.fr

Hichem Snoussi
University of technology of Troyes
Troyes, France
hichem.snoussi@utt.fr

Mahmoud Soua
AQUILAE
Troyes, France
m.soua@aquilae.tech

***Abstract*—Unsupervised anomaly in industry has been a concerning topic and a stepping stone for high performance industrial automation process. The vast majority of industry-oriented methods focus on learning from good samples to detect anomaly notwithstanding some specific industrial scenario requiring even less specific training and therefore a generalization for anomaly detection. The obvious use case is the fabric anomaly detection, where we have to deal with a really wide range of colors and types of textile and a stoppage of the production line for training could not be considered. In this paper, we propose an automation process for industrial fabric texture defect detection with a specificity-learning process during the domain-generalized anomaly detection. Combining the ability to generalize and the learning process offer a fast and precise anomaly detection and segmentation. The main contributions of this paper are the following: A domain-generalization texture anomaly detection method achieving the state-of-the-art performances, a fast specific training on good samples extracted by the proposed method, a self-evaluation method based on custom defect creation and an automatic detection of already seen fabric to prevent re-training.**

***Index Terms*—Domain-Generalization, unsupervised, anomaly, unseen, knowledge distillation, student-teacher, memory banks, fabric, automation.**

I. Introduction

Unsupervised anomaly detection in industry is a vast topic, since there are a lot of possible applications. In this paper, we focus on fabric anomaly, which is a concerning topic for industry. The specificity of fabric is the pattern in the structure and if we manage to understand that pattern we can extract anomalies. Several methods have been introduced for industry anomaly detection using MVTEC AD [1] the dataset that gathers textures (carpet, leather, grid, wood, and tile) and objects (bottle, cable, capsule, hazelnut, metal nut, pill, screw, toothbrush, transistor and zipper). These methods could achieve high performance. However, they rely on object/texture specific unsupervised learning without generalization capacity. Recently, knowledge-distillation based methods have been introduced for the unsupervised anomaly detection task [2]. It consists of a student-teacher model focusing on the bottom layers of the network as they represent the edges, color and shapes information. We used the same approach to design a domain-generalized texture anomaly detection method with the ability to detect defects on unseen textures and to select good samples for a texture-specific unsupervised anomaly detection model. In fabric industry, many types and colors of fabric are analyzed, and it would be impossible to rely on a specific training on good samples for each type of fabric without slowing the industrial process.
Therefore, we propose a complete data processing chain for a robust, fast and adaptive texture specific anomaly detection and localization. Our method is based on four main modules: a domain-generalized texture anomaly detector, a fast texture specific training/inference, an auto-evaluation process of our specific model and an automatic already-seen fabric detection to avoid retraining an existing model.
The paper is organized as follows. In section II, we review the related work especially on MVTEC dataset and present the different approaches proposed in literature for domain-generalized and classic unsupervised anomaly detection. In section III, we present an enhanced domain-generalized texture defect detection method. In section IV, we present the specific learning method, the auto-evaluation process and the already seen texture recognition. Section V is dedicated to the analysis of the results. Section VI concludes the paper.

II. Related works

As our proposed methods address two specific tasks, we first present the state of the art on domain-generalized texture anomaly detection and then the state of the art on unsupervised defect detection of known objects.

A. Domain-generalized texture anomaly detection

Domain-generalized anomaly detection is an important topic for optimal industrial process, since in specific industrial fields, the type of textures often changes. The most obvious example is certainly fabric anomaly detection where fabric can have different colors (red, blue, striped) and types (cotton, polyester, silk, etc). The main objective is to detect defects on any type of fabric without resorting to a time-consuming training. The feature extraction from a pretrained classifier offers the most promising results with different types of networks such as an episodic training [3], the use of extrinsic and intrinsic aspects [4] and multiscale feature extractor with co-attention modules [5].

B. Unsupervised anomaly detection on known objects

More commonly, unsupervised anomaly detection deals with the problem of detecting defects on an object or texture based on only good samples. In industry or security scenarios, we often have a low rate of defects with a vast number of different defect types which would lead to a time-consuming annotation and a possibly non-pertinent classification if all the anomaly types are not considered[6]. To tackle this question, several methods emerged proposing different types of algorithms such as autoencoders [7] and variational autoencoder variants [8] [9]. Another common way of detecting anomalies is Generative Adversarial Networks (GAN) introduced by [10] adapted to unsupervised anomaly detection such as AnoGAN [11], G2D [12] and OCR-GAN [13]. More recently, approaches using a pretrained classifier has been at the heart of the research in industrial anomaly detection and offers outstanding performance. There are three main feature extraction-based approaches: normalizing flow, knowledge distillation and memory banks. The normalizing flow approach consists of a flow training based on relevant features of good samples from a pretrained network such as AlexNet [14], Resnet [15] or efficient-net [16] trained on imageNet. Different strategies were used to enhance performance, such as a 2D flow [17] or a cross-scale flow [18]. Another interesting approach is the use of a memory bank to extract relevant information from different good samples and to use this memory bank to compare and detect if there is an anomaly [19]. Finally, the concept of knowledge distillation was adapted for unsupervised anomaly detection and localization [2]. The idea is to train a student network based on the output features of a teacher (already pretrained for a classification purpose) and on good samples. The student will be able to reproduce teacher features on a good sample, but will not be as precise for a defective sample. Several methods used this principle with different strategies such as a multi-layer feature selection [2], an asymmetric student teacher [20], a coupled-hypersphere-based feature adaptation [21] and a mixed-teacher approach [22].

III. Knowledge distillation generalization

The proposed model is based on the knowledge distillation framework, where a pretrained network is used as a teacher and a student network is trained to reproduce the teacher output on good samples. The student network is then expected to not be able to reproduce teacher features on defective samples, a property which is used to detect abnormal samples. For domain generalization, we propose to train the student on different types of textures and using many teachers to guarantee generalization. In order to achieve this objective, we first constitute a new dataset based on fabric datasets [23] which regroups different categories of textures with different quality and homogeneity.

Then, to tackle the problem of texture domain generalization, we used a specific student teacher architecture with different branches based on the paradigm that each pretrained classifier have a different bias towards classification.

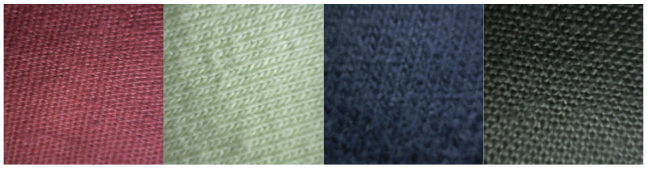

Fig. 1: Samples employed for the custom fabric dataset (extracted from the fabrics dataset [23])

In terms of layer selection, the deeper a layer, the more the information relates to the context and conversely, the shallower a layer, the more information it contains on contours, edges, and colors. Based on different layer configurations, we show that for the purpose of texture domain generalization, mid-level features would be the best choice to combine texture specific information such as contours and edges and a general vision of what a texture is.

At least two classifiers are needed to attenuate each bias. We have used Resnet18 and EfficientNet-b0 for computation time speed and meaningful features.

To fully exploit each classifier information, we used a parallel architecture which can be seen as a multiple teachers/multiple students architecture where the training happen independently for each classifier, only the anomaly score is calculated with the two networks outputs. Our framework is an adaptation of MixedTeacher [22] with a different layer selection strategy. The first Resnet layer is not used as its output features are too specific to training dataset textures. We used the features of the three first residual blocks of Resnet18 and the last 2 convolutional blocks of efficientNet-b0. As in [22], we used a reduced version of the Resnet18 model with a reduction of the block size and a reduction of the dimension of each layer with an adaptive average pooling, while we keep the same architecture for the EfficientNet part.

Given a training dataset of images without anomaly $D = [I_1, I_2, ..., I_n]$, our goal is to extract the information of L mid-level layers. For an image $I_k \in R^{w*h*c}$ where w is the width, h the height, and c the number of channel, the teacher outputs features $F_t^l(I_k) \in R^{w_l*h_l*c_l}$ and $F_s^l(I_k) \in R^{w_l/2*h_l/2*c_l/2}$ with $l > 1$ and $F_s^l(I_k) \in R^{w_l*h_l*c_l}$ if $l = 1$. The loss is obtained by applying the $l2$ distance of normalized feature vectors for each pixel of the feature map and summing them. For the Resnet student part, we used an adaptive average pooling layer on teacher features. The used layers are $l = \{1, 2, 3\}$ for the Resnet part and $l = \{5, 6\}$ for the EfficientNet part.

Pixel loss for the resnet part is defined in the following Eq.1:

$$loss^l(I_k)_{ij} = \frac{1}{2}\|norm(AAP(F^l_{Resnet18}(I_k))_{ij}) - norm(F^l_s(I_k)_{ij})\| \quad (1)$$

where AAP refers to Adaptive Average Pooling. For the EfficientNet part, pixel loss is defined in the following Eq.2:

$$loss^l(I_k)_{ij} = \frac{1}{2}\|norm(F^l_{EffNetb0}(I_k)_{ij}) - norm(F^l_s(I_k)_{ij})\| \quad (2)$$

For the layer l, the loss is defined as:

$$loss^l(I_k) = \frac{1}{w_l h_l}\sum_{i=1}^{w_l}\sum_{j=1}^{h_l} loss^l(I_k)_{ij} \quad (3)$$

and finally, for the total loss is written as:

$$loss(I_k) = \sum^{l} loss^l(I_k) \quad (4)$$

IV. Auto-learning process for industrial deployment

The previous part was presented in the context of industrial efficiency, where it was not allowed to retrain for every new type/color of texture/fabric. The objective of this section is to propose a general classifier for handling the anomaly detection role while we gather enough images and train a specific model for increased efficiency.
This section is divided in 3 parts: training and self-evaluation, recognition of an already trained type of fabric, and a typical industrial use-case in fabric industry.

A. Training and self-evaluation

Given the deployment constraints, we considered different criteria for the choice of the student-teacher network architecture: (i) the inference and training time, (ii) the performance and (iii) the robustness to defective samples in the training set. We also considered the possibility of running the process on several asynchronous defect detectors. The model Reduced Student proposed in [22] is a good candidate. Thanks to its reduced architecture, we can train a specific model in an acceptable time. To minimize the number of potential defective samples in the training, we gathered the samples with acceptable anomaly score from the domain-generalized model, i.e samples classified as good samples. Based on a test-error approach, we determined the optimal number of epochs (during specific training) where the specific model becomes better than the domain-generalized one so that we can start using the best model even if the complete training is not finished.
The self-evaluation part is based on two types of data: (i) the first type is defective samples detected by the domain-generalized anomaly detector and (ii) the second type is generated data with a procedure inspired by DRAEM [9]: Perin noise and the texture database dtd [24]. We used the same approach to generate non-absurd defects to self-evaluate our model.

B. Already seen fabric recognition

To guarantee an automated anomaly detector without the help of an operator for selecting an already-trained model, we propose an algorithm to precisely recognize a fabric type already considered previously. For each trained model, we save x extracted features from the specific model on good samples reduced using coreset subsampling introduced in PatchCore [19] to guarantee fast computation. Each specific model is also saved in a model bank of N models and linked to its features in a feature bank. When we have to decide if the fabric was already seen, we calculate the sample/model proximity by extracting features from all trained specific models from the model bank, applying the coreset subsampling and comparing these features to the x features from the feature bank of each specific model with cosine similarity distance as described in equation 6. We then compute the intra-class proximity by calculating the cosine similarity between the x features of the same model as reported in equation 7. The proximity score is defined as the absolute value of the difference between the sample/model proximity and the intra-class proximity. We finally make the decision by comparing the maximum proximity score with a $similarityThreshold$. The threshold is chosen based on what is known about the similarity between the fabric.

Even though it may seem laborious if the model bank is consistent, it is still real-time deployable thanks to the inference speed of the reduced student architecture proposed in the previous subsection and the coreset subsampling, as we show in the experiment part. This is by far the most accurate method for comparing a new piece of fabric with a previously seen one and, we believe, it is still usable even in a specific case of thousands of specific models.
The cosine similarity formula is:

$$sim(feat_A, feat_B) = \frac{feat_A.feat_B}{\|feat_A\|\,\|feat_B\|} \quad (5)$$

with $feat_A$ and $feat_B$ the extracted features. The sample/model proximity is defined as:

$$prox_{sm}(S, Model) = \frac{1}{x}\sum_{i=1}^{x} sim(feat_S, feat_i) \quad (6)$$

The intra-class proximity is defined as:

$$prox_{ic}(Model) = \frac{1}{x(x-1)}\sum_{i=1}^{x}\sum_{j=1, i\neq j}^{x} sim(feat_i, feat_j) \quad (7)$$

The proximity score is :

$$proxScore(S, Model) = abs(prox_{sm}(S, Model) - prox_{ic}(Model)) \quad (8)$$

And the already-seen decision is described as :

$$\max_{i\in N}(proxScore(S, Model_i)) > similarityThreshold \quad (9)$$

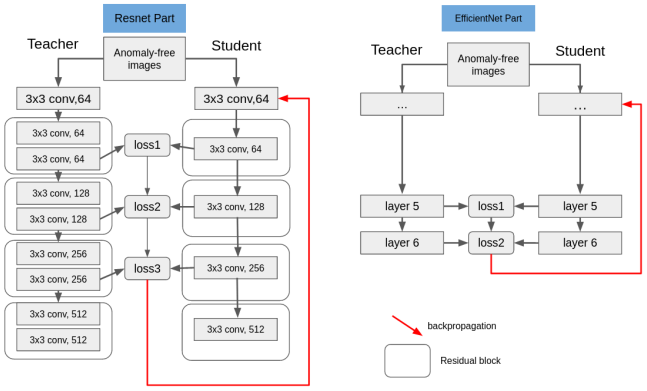

Fig. 2: Architecture of Resnet student teacher (left) and EfficientNet student teacher (right)

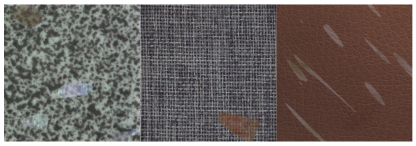

Fig. 3: Custom defective samples generated with Perin noise

C. Typical use-case : fabric industry

To demonstrate the effectiveness of our method, we describe a typical real defect analysis use case. In a vast majority of mid-range clothing industry, the fabric is analyzed several times during the whole fabrication process by operators that scroll the fabric and look for defects. This is a laborious job and often distraction occurs resulting in a globally low detection percentage of defects, not to mention the difficulty for the eyes to look at certain fabric categories such as striped fabrics. Our automated process aims at assisting the operator for the classification task and to speed up the scrolling of the fabric. The operator is still needed since he has to install the fabric roll on the machine and to verify the defect classification done by the domain generalized model since the accuracy is still low for a full automation process.

For every fabric roll, the process start with an identification of the fabric to control. Two different cases may happen:

- If this type of fabric has never been seen, a specific training is started while still doing the anomaly detection with the domain-generalized model, we may have to slow the scrolling of the fabric during the training depending on the computational power. When the trained model becomes better than the domain-generalized model, we used it instead, even if the training is not completely finished. When the training is finished, we used the completely trained model for anomaly detection while keeping some features of defective samples for the recognition part.
- If this type of fabric is already-seen, the specific trained model is used for anomaly detection.

The process is fully automated and does not require any help from the operator except for the activation or deactivation, which could be done also by connecting the visiting machine with the central unit to send an activation signal.

V. Experiments

This section is divided into 3 parts: the analysis of the domain-generalization model compared with state of the art for different training configurations, the analysis of the training speed and inference speed of our model and finally the estimation of the number of required epochs on a specific training to outperform the domain-generalization algorithm.

A. State-of-the-art comparison

For the evaluation of our model, we used two different databases for training. For the “MVTEC” one, we trained the DG model on all good samples of MVTEC AD textures except the one we are testing on to reproduce the evaluation protocol of the other state-of-the-art papers. The “cotton” one is trained on the custom fabric dataset presented in section III and was created for fabric anomaly which explain the SOTA performances on carpet and leather. The results are presented in table I.

For the training, we used stochastic gradient descent with a learning rate of 0.4 for 200 epochs with a batch size of 16. Both networks are pretrained on ImageNet. We resized all the images to 256x256, keeping 80% for training and 20% for validation. We kept the checkpoint with the lowest validation loss.

TABLE I: **AUC comparison between our method and existing ones on MVTEC AD**

textures	Epi-FRC+[3]	EISNet+[4]	DGTSAD[5]	Ours(MVTEC)	Ours(Coton)
carpet	0.916	0.982	0.943	0.985	**0.996**
leather	1.000	1.000	**1.000**	0.991	0.996
wood	0.941	0.979	0.962	**0.999**	0.948
tile	0.951	0.851	**0.994**	0.965	0.964
grid	0.725	0.728	0.730	0.937	**0.944**
mean	0.907	0.909	0.918	**0.975**	0.969

As seen in table I, the 2 types of training offers approximately the same mean AUC but the dataset “cotton” only contains one hundred images and is supposed to be effective for fabric defects detection whereas it shows the best results

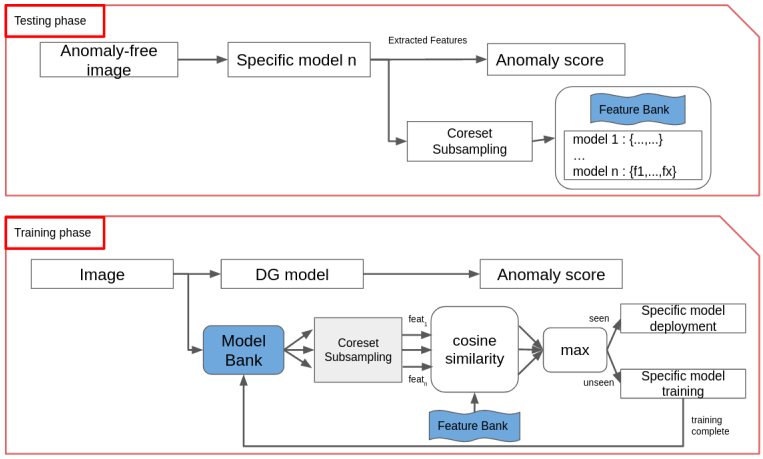

Fig. 4: Fabric Recognition process : The extraction of anomaly-free images features in the testing phase stops when the x number of elements is reached, the model bank is the bank of all previously trained specific models

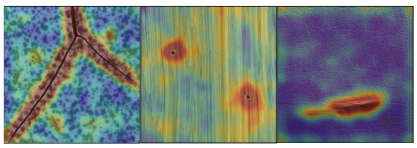

Fig. 5: Outputs of detected anomaly with our domain-generalization model

on both carpet, leather and grid which contains patterns [25] and are the most fabric-like textures of the dataset MVTEC AD.

Comparing to the state-of-the-art methods, our approach obtains 0.057 AUC more than the previous best model, which is an excellent improvement and is in the way for closing the gap between domain generalization and classical anomaly detection for textures.

B. Inference speed

For the inference speed, all the following tests were done with a RTX 2080Ti. The training parameters are the same as the previous part except for the number of epochs where we limited it to 100. To perform these experiments, we used an optimal algorithm for fast processing in batch of 8 which outperforms the classic anomalib [26] in terms of inference speed on knowledge distillation methods for anomaly detection (see table II). For the training time, with 100 epochs and 100 images, we report 4 minutes and 20 seconds.

TABLE II: **Inference speed**

Category	Ours(Domain-Generalized)	Ours(Specific)
FPS	333	**1111**
Latency (ms)	3	**0.9**

C. Necessary training epochs before model replacement

The main purpose of the whole method is to detect defects as precisely as possible during the whole process. We need to estimate at which point the specific model surpasses the DG model in terms of AUC. In order to find this number, we tested our specific model at different epochs, and we compared results to the DG model performance.

TABLE III: **Epoch performance**

textures	10 epochs	20 epochs	30 epochs	100 epochs	DG model
carpet	0.915	0.920	0.963	1.0	0.996
leather	0.949	0.974	0.990	0.997	0.996
wood	0.988	0.989	0.995	0.996	0.948
tile	0.986	0.987	0.991	0.987	0.964
mean	0.959	0.967	0.984	0.995	0.969

Based on the mean results, we could consider using the specific model after 30 epochs of training i.e one minute and 30 seconds, but considering the scores of carpet and leather which are the most fabric-like textures, it may be a counter performance to switch to the specific model before the end of the training if the DG model is close enough to real distribution.

VI. Conclusion and Future Work

We proposed an industry-ready automation process for industrial defect detection, especially deployable for fabric with fast inference results for both domain-generalized and specific model. The outstanding capability of our architecture to mutual aid between humans and artificial intelligence makes it a great and reliable tool for visual inspection. Nevertheless, several improvements could be considered. According to [6], semi-supervised can easily surpass unsupervised anomaly detection even with a few annotated anomalies and this could be applied to our method by using the DG detected anomalies to train a semi supervised specific model to increase the detection performances. Some automation improvement could be considered in terms of potential use of the method, such as a direct software included in the fabric visiting machine to monitor speed, stop and dysfunction mode more precisely.

References

[1] P. Bergmann, M. Fauser, D. Sattlegger, and C. Steger. "MVTec AD — A Comprehensive Real-World Dataset for Unsupervised Anomaly Detection". In: *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). Long Beach, CA, USA: IEEE, June 2019, pp. 9584–9592.

[2] G. Wang, S. Han, E. Ding, and D. Huang. "Student-Teacher Feature Pyramid Matching for Anomaly Detection". In: *arXiv:2103.04257 [cs]* (Oct. 28, 2021).

[3] D. Li, J. Zhang, Y. Yang, C. Liu, Y.-Z. Song, and T. Hospedales. "Episodic Training for Domain Generalization". In: *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*. 2019 IEEE/CVF International Conference on Computer Vision (ICCV). Seoul, Korea (South): IEEE, Oct. 2019, pp. 1446–1455.

[4] S. Wang, L. Yu, C. Li, C.-W. Fu, and P.-A. Heng. *Learning from Extrinsic and Intrinsic Supervisions for Domain Generalization*. July 17, 2020.

[5] S.-F. Chen, Y.-M. Liu, C.-C. Lin, T. P.-C. Chen, and Y.-C. F. Wang. *Domain-Generalized Textured Surface Anomaly Detection*. Mar. 23, 2022.

[6] S. Han, X. Hu, H. Huang, M. Jiang, and Y. Zhao. *ADBench: Anomaly Detection Benchmark*. Sept. 16, 2022.

[7] S. Mei, Y. Wang, and G. Wen. "Automatic Fabric Defect Detection with a Multi-Scale Convolutional Denoising Autoencoder Network Model". In: *Sensors* 18.4 (Apr. 2018), p. 1064.

[8] Q. P. Nguyen, K. W. Lim, D. M. Divakaran, K. H. Low, and M. C. Chan. "GEE: A Gradient-based Explainable Variational Autoencoder for Network Anomaly Detection". In: *arXiv:1903.06661 [cs, stat]* (Mar. 15, 2019).

[9] V. Zavrtanik, M. Kristan, and D. Skočaj. *DRAEM – A discriminatively trained reconstruction embedding for surface anomaly detection*. Sept. 27, 2021.

[10] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. "Generative Adversarial Networks". In: *arXiv:1406.2661 [cs, stat]* (June 10, 2014).

[11] T. Schlegl, P. Seeböck, S. M. Waldstein, G. Langs, and U. Schmidt-Erfurth. "f-AnoGAN: Fast unsupervised anomaly detection with generative adversarial networks". In: *Medical Image Analysis* 54 (May 2019), pp. 30–44.

[12] M. Pourreza, B. Mohammadi, M. Khaki, S. Bouindour, H. Snoussi, and M. Sabokrou. "G2D: Generate to Detect Anomaly". In: *2021 IEEE Winter Conference on Applications of Computer Vision (WACV)*. 2021 IEEE Winter Conference on Applications of Computer Vision (WACV). event-place: Waikoloa, HI, USA. IEEE, Jan. 2021, pp. 2002–2011.

[13] Y. Liang, J. Zhang, S. Zhao, R. Wu, Y. Liu, and S. Pan. "Omni-frequency Channel-selection Representations for Unsupervised Anomaly Detection". In: *arXiv:2203.00259 [cs]* (Mar. 1, 2022).

[14] K. He, X. Zhang, S. Ren, and J. Sun. *Deep Residual Learning for Image Recognition*. Dec. 10, 2015.

[15] A. Krizhevsky, I. Sutskever, and G. E. Hinton. "ImageNet classification with deep convolutional neural networks". In: *Communications of the ACM* 60.6 (May 24, 2017), pp. 84–90.

[16] M. Tan and Q. V. Le. *EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks*. Sept. 11, 2020.

[17] J. Yu, Y. Zheng, X. Wang, W. Li, Y. Wu, R. Zhao, and L. Wu. "FastFlow: Unsupervised Anomaly Detection and Localization via 2D Normalizing Flows". In: *arXiv:2111.07677 [cs]* (Nov. 16, 2021).

[18] M. Rudolph, T. Wehrbein, B. Rosenhahn, and B. Wandt. "Fully Convolutional Cross-Scale-Flows for Image-based Defect Detection". In: *arXiv:2110.02855 [cs]* (Oct. 6, 2021).

[19] K. Roth, L. Pemula, J. Zepeda, B. Schölkopf, T. Brox, and P. Gehler. "Towards Total Recall in Industrial Anomaly Detection". In: *arXiv:2106.08265 [cs]* (June 15, 2021).

[20] M. Rudolph, T. Wehrbein, B. Rosenhahn, and B. Wandt. *Asymmetric Student-Teacher Networks for Industrial Anomaly Detection*. Oct. 18, 2022.

[21] S. Lee, S. Lee, and B. C. Song. *CFA: Coupled-hypersphere-based Feature Adaptation for Target-Oriented Anomaly Localization*. June 9, 2022.

[22] S. Thomine, H. Snoussi, and M. Soua. "MixedTeacher : Knowledge Distillation for fast inference textural anomaly detection". In: VISAPP international conference on computer vision theory and applications. Feb. 19, 2023.

[23] C. Kampouris, S. Zafeiriou, A. Ghosh, and S. Malassiotis. "Fine-Grained Material Classification Using Micro-geometry and Reflectance". In: *Computer Vision – ECCV 2016*. Ed. by B. Leibe, J. Matas, N. Sebe, and M. Welling. Vol. 9909. Series Title: Lecture Notes in Computer Science. Cham: Springer International Publishing, 2016, pp. 778–792.

[24] M. Cimpoi, S. Maji, I. Kokkinos, S. Mohamed, and A. Vedaldi. "Describing Textures in the Wild". In: *2014 IEEE Conference on Computer Vision and Pattern Recognition*. 2014 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). Columbus, OH, USA: IEEE, June 2014, pp. 3606–3613.

[25] H. Y. T. Ngan, G. K. H. Pang, and N. H. C. Yung. "Automated fabric defect detection—A review". In: *Image and Vision Computing* 29.7 (June 1, 2011), pp. 442–458.

[26] S. Akcay, D. Ameln, A. Vaidya, B. Lakshmanan, N. Ahuja, and U. Genc. *Anomalib: A Deep Learning Library for Anomaly Detection*. Feb. 16, 2022.