Journal of Computer Science

Original Research Paper

DifferSqueezeNet: A Leaner Defect Detection Model

Jose Joao Manrique Franco and Marcelo Rudek

Industrial and Systems Engineering Graduate Program, Pontifical Catholic University of Parana, Curitiba, Brazil

Article history
Received: 08-06-2024
Revised: 09-08-2024
Accepted: 04-10-2024

Corresponding Author:
Jose Joao Manrique Franco
Industrial and Systems Engineering Graduate Program, Pontifical Catholic University of Parana, Curitiba, Brazil
Email: franco.jose@pucpr.edu.br

Abstract: In the industrial environment, the inherent limitations of manual inspection have caused the ascent of automated computer vision systems that aim to match or surpass human performance. In this context, the artificial intelligence field has been interested in defect detection with the creation of many machine learning techniques, focusing on Unsupervised and Semi-supervised learning methods. Since most of these methods use large models in their architectures, we propose a new model architecture that aims to adapt an already existing architecture into a leaner one. We present DifferSqueezeNet, a model that not only is smaller in size but also improves its baseline architecture, delivering better performance at image-level anomaly detection while consuming less computational resources.

Keywords: Anomaly Detection, Computer Vision, Deep Learning, Visual Inspection

Introduction

Humans are accustomed to observing things throughout their lives, often without much interest. However, sometimes, something grabs an individual's attention, signaling an opportunity or a potential danger. This ability to recognize something new or different, deviating from familiar patterns, highlights a distinct human trait.

This phenomenon finds its equivalent in the academic field of anomaly detection. This field spans diverse domains, from ferreting out credit card fraud and identifying cyber intrusions to the vigilant monitoring of equipment performance (Chandola *et al*., 2009; Kakavand *et al*., 2015).

In the current era of advancing deep neural networks, the field of anomaly detection has expanded its applications to multimedia domains such as sound and imagery, making the subject gain significant importance. For example, this broader scope includes applications in medical diagnoses, surface defect detection, and intrusion alerts (Ye *et al*., 2020).

In the industrial environment, the visual inspection process ensures high production quality and cost efficiency by the action of discarding defective parts. When assembly lines manufacture many instances of products, most are normal and fault-free. Yet, on some occasions, the manufactured products might contain some faults that come in different types, like cracks, cuts, or holes. Examples of this type of defect can be seen in Fig. (1).

However, since manual inspection by humans is beset by sluggishness, expense, and potential fallibility, the use of an automated computer vision system is becoming a popular topic in this research field. This shift has paved the way for the creation of a plenitude of systems aiming to surpass human performance in these critical scenarios.

Using this inspiration, the artificial intelligence research field has been focusing on defect detection with the development of machine learning techniques, most of them using Convolutional Neural Networks (CNNs) architectures. The adaptability of CNNs' filters to specific problem contexts and their expertise in extracting intricate patterns from two-dimensional images have put them in evidence.

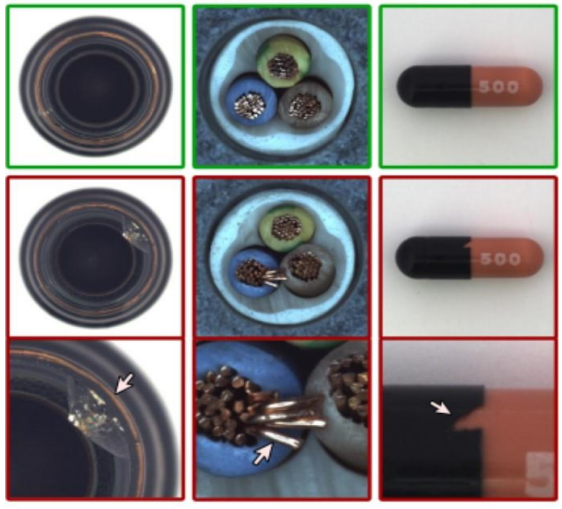


Fig. 1: Normal and defective manufactured products. The defect is indicated by the white arrow. Adapted from (Bergmann *et al*., 2019)

Problem Statement

Frequently quality inspection is an integral part of the production process, requiring constant monitoring and improvement. Hence, small defects during fabrication need to be detected reliably, in order to assure production quality and prevent incidents.

However, it is difficult to know in advance which types of defects can occur. These events are not so frequent and even if they are already known, new types that were not previously observed can still happen.

Therefore, some defect detection applications are unable to use common supervised machine learning approaches, due to the need for labels in every registry of the training data. Thus, it is a common effort in the research field to look for alternatives to identify defects without having to know all the different types that can exist. For this purpose, some unsupervised learning methods and semi-supervised ones are being utilized, due to their ability to learn and recognize patterns with few samples of annotated data, like the ones presented by (Rudolph *et al*., 2021; Defard *et al*., 2021).

However, many well-established defect detection model architectures present large and computationally intensive models in their architectures, like ResNet (He *et al*., 2016) or AlexNet (Krizhevsky, 2014). This can pose significant challenges, especially in some industrial environments that often work with limited computational resources.

Contribution of the Paper

To mitigate the aforementioned issues, we propose a new defect detection method that aims to adapt the DifferNet (Rudolph *et al*., 2021) model architecture into a new one with a leaner architecture. The following guidelines guide our approach:

- Propose an architecture that presents a substantial reduction in model size and can run inference in modest computational devices
- Compare the results of the new model architecture and see if it is equivalent or better than the baseline model
- Verity if the proposed model architecture presents a competitive to the state-of-the-art performance in the task of defect detection

Related Work

Defect Detection is motivated by the importance of quality control in the industrial production line. The manufacturing environment is inherently susceptible to various types of defects and the emergence of new defect types underscores the need for a versatile detection approach that doesn't rely on prior knowledge of all possible defects.

This scenario has driven the convergence of research in the fields of semi-supervised and unsupervised learning with defect detection. Numerous studies have emerged from this synergy, offering innovative approaches to tackle the challenge. These approaches can be broadly categorized into either reconstruction-based or embedding similarity-based methods, depending on the logic used to learn the data patterns and how to generate an anomaly score.

Reconstruction-Based Methods

These kinds of models are widely used for anomaly detection and localization and employ architectures like Autoencoders (AE) (Fauser *et al*., 2019; Ye *et al*., 2020), Variational Autoencoders (VAE) Venkataramanan *et al*., 2020; Liu *et al*., 2020) or the Generative Adversarial Networks (GANs) (Sabokrou *et al*., 2018; Akcaymir *et al*., 2019) are trained to reconstruct normal (non-anomalous) images only.

Therefore, the model relies on the fact that abnormal images are not well reconstructed by the GAN reconstruction process. Thus, the anomaly score for a sample is given by the quality of a reconstructed image, comparing the model input to its output.

If the difference between the two images is significant, the image is flagged as an anomaly. This difference is measured by a metric called reconstruction error (Ye *et al*., 2020) and is usually defined as a loss function. For example, Akcaymir *et al*. (2019) defined the anomaly score as:

$$A(x) = ||GE(x) - E(G(x))|| \quad (1)$$

where, $G_E(x)$ represents the input image features and $E(G(x))$ stands for the encoded features of the generated image.

However, sometimes can occur that some autoencoder based methods fail to perform due to high reconstruction generalization, causing even anomalies to be reconstructed as well as normal samples, a behavior that implies low anomaly scores.

Embedding Similarity-Based Methods

These methods use CNN models that are trained on large-scale datasets like Imagenet (Krizhevsky *et al*., 2017) to extract meaningful feature vectors and then perform a comparison in the distributions in order to determine if an image contains an anomaly or not.

This is usually presented in an anomaly score, which is generally calculated by the distance between the embedding vectors of the image being tested and the reference vectors the model generated in the training phase.

In particular, Cohen and Hoshen (2020) defined the anomaly score as the Euclidean distance between the test

image y and the K nearest normal image from the training set, $N_k(f_y)$. So, the distance is presented as:

$$d(y) = \frac{1}{k} \sum_{f \in N_i(f_i)} \| f - f_y \|^2 \quad (2)$$

Some methods are image-level, using the model to describe an entire image for anomaly detection (Bergman and Hoshen, 2020; Siddiqui *et al*., 2018; Bergman *et al*., 2020; Hu and Wang, 2022) or sub-image level, using an image patch for anomaly localization (Yi and Yoon, 2021; Cohen and Hoshen, 2020; Napoletano *et al*., 2018).

State-of-the-Art

The current state-of-the-art is PatchCore (Roth *et al*., 2022) which uses a pre-trained CNN model and a memory bank of feature representations in its architecture. The anomaly score is calculated through the nearest nearest-neighbor search.

Another relevant model is CFLOW-AD (Gudovskiy *et al*., 2022) which uses a Conditional Normalizing Flow (Winkler *et al*., 2019) framework as a decoder, enabling the model to learn the distribution of anomaly-free images. It calculates the anomaly score through likelihood estimation.

Finally, we have the PaDiM (Defard *et al*., 2021) model which uses a pre-trained CNN for embedding extraction and describes each patch position by a multivariate Gaussian distribution in the training stage. It calculates the anomaly score using the Mahalanobis distance.

Memory Efficiency in Defect Detection Models

Memory efficiency is not commonly emphasized in many defect detection models; however, certain models have attempted to address this issue, exemplified by PaDiM (Defard *et al*., 2021). This study experimented with two different Feature Extractors, one using the regular Resnet-50 CNN and the other using a Resnet-18 (He *et al*., 2016). The results reported by the authors demonstrated that it could be possible to have less memory complexity without losing significant performance in the task of defect detection (measured by the AUROC metric).

Another recent work that approached this subject was CFLOW-AD (Gudovskiy *et al*., 2022), which presented experiments with multiple Feature Extractors, such as Resnet-50, Resnet-18 (He *et al*., 2016) and MobileNetV3 (Howard *et al*., 2019). The model presented decent performance with the smaller FEs both in the tasks of defect detection and localization.

The complexity of a defect detection model relies on the image resolution of the dataset and the size of the networks used in the architecture. If it is an embedding similarity-based architecture, the FE could play an important role in the total size of the model. (Defard *et al*., 2021).

The DifferNet Model

This model, presented in the paper by Rudolph *et al*. (2021), introduces an embedding-similarity approach that contains two key components in the model architecture: The Feature Extractor (AlexNet) and the Normalizing Flows (NF) network.

The Feature Extractor is responsible for extracting relevant information from the input images, while the NF works to convert this input into a normal distribution using maximum likelihood training. Consequently, this model architecture relies on the NF network's capability to detect Out-of-Distribution (OOD) samples.

The different models used a model architecture of coupling layers as proposed in the Real-NVP structure (Dinh *et al*., 2016) in the Normalizing Flow network. In the NF, we can resize the number of coupling blocks of the network and the Number of Neurons in the Hidden Layers to refine our model architecture.

Materials and Methods

In this study, we have created a lean defect detection model with the help of a given dataset that is a benchmark for anomaly detection with a focus on industrial inspection.

Materials

The MVTec AD dataset (Bergmann *et al*., 2019) is a publicly accessible multi-defect and multi-object anomaly dataset. It encompasses a repository of 5,354 high-resolution color images representing 10 objects and 5 distinct texture classes. The dataset comprises 3269 images for training and 1,725 for testing.

Being a dataset for both defect detection and localization tasks, the MvTec AD dataset also provides a pixel-accurate ground truth region for each defect image (1,888 in total). Due to these features, this dataset is widely adopted by works that approach anomalies in the industrial visual inspection context, such as Defard *et al*. (2021); Gudovskiy *et al*. (2022); Napoletano *et al*. (2021).

Methods

With the dataset in hand, we needed to perform some training and evaluation steps to guide the decision-making of our research. During the training process, the model is only exposed to normal images. Defective images are only used in the evaluation phase.

Thus, aligned with our research objective of creating a leaner model based on DifferNet (Rudolph *et al*., 2021) and knowing that a Feature Extractor is important to the memory complexity of a defect detection architecture, we decided that the genesis of our new architecture would involve adopting lean FEs. This was reinforced by the findings of the different papers, which hinted at a limited use of pre-trained CNN as FEs in their tests.

Thus, experiments were conducted using a SqueezeNet (Iandola *et al*., 2016) pre-trained model as a

feature extractor to address this research gap. Those prior tests showed that the chosen FE did not significantly harm the model's performance, which was gauged by the AUROC metric and stayed at a similar level.

Consequently, as we delved into the baseline's architectural intricacies, we identified another three characteristics that could be parameterized:

- Number of Features: the dimension of the features that are collected from the feature extractor and passed as input to the NF network
- Number of Coupling blocks: the number of blocks presented in the NF network
- Number of Neurons in the Hidden Layers: number of neurons used in the fully connected layers nested within the Normalizing Flows blocks

Thus, we proceeded to an iterative experimentation phase, in which we tried different types of settings changing the parameterized characteristics. The training process unfolded on an Nvidia GeForce RTX 3080 GPU, a computational resource that facilitated efficient experimentation. In this process, we used a Random Search (Bergstra and Bengio, 2012) to iterate through the parameters.

Through the aforementioned iterative refinement process, the alterations introduced have yielded substantial advancements in model performance, demonstrating a significant leap when compared to the baseline model architecture. A glance at the architecture of the final model can be seen in Fig. (2).

Due to the relevance of the SqueezeNet feature extractor in the model architecture and to reference the baseline model, we decided to name it DifferSqueezeNet.

Ultimately, the experiments involving adjustments to the parameters "Number of Coupling Blocks" and "Number of Neurons in the Hidden Layers" did not yield favorable results. Reducing these values had a detrimental effect on the model's performance while increasing them failed to produce significant improvements in the AUROC metric but increased model complexity.

The optimal parameter configuration obtained after the iterative process is presented in Table (1) in comparison to the parameters originally defined by the baseline model.

Just like in the baseline model, the anomaly score is calculated by the average of the negative loglikelihoods using multiple transformations $\tau_i \in T$ of an input image x:

$$\tau(x) = E\tau i \in T[-log\, pZ\ (fNF(fex(\tau(x))))] \quad (3)$$

where, T represents rotations and manipulations of brightness and contrast. An image is classified as anomalous if the anomaly score $\tau(x)$ is above the threshold value θ.

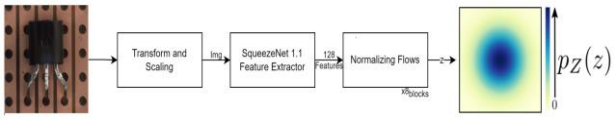


Fig. 2: Overview of the DifferSqueezeNet architecture

Table 1: Comparison between the parameters used in the original DifferNet model and our approach, DifferSqueezeNet

Parameter	DifferNet	DifferSqueezeNet
Feature Extractor	AlexNet	SqueezeNet v1.1
Training Epochs	24	24
Number of Features	256 * 3 scales	128 * 3 scales
Number of Neurons in Hidden Layer	2048	2048
Number of Coupling Blocks	8	8

A key observation is the performance demonstrated by our custom-crafted model, achieving commendable results while embracing a more streamlined and efficient model architecture compared to the original baseline. This achievement attests to our dedication to maximizing performance within practical constraints, an important consideration for many real-world deployment scenarios.

For instance, edge computing is particularly relevant to our refined model's practical applicability. In edge computing, data processing and computation occur closer to the data source, reducing latency and enabling real-time decision-making at the network edge. This model's more modest architecture aligns coherently with the resource-constrained environments often encountered at the edge, making it an optimal choice for scenarios where computational resources are limited.

The next section will present the details of the model's performance outcomes, highlighting the achievements and implications of our refined model architecture. These results present the culmination of the research, showcasing the positives and negatives of our new proposal.

Results

This section provides a comprehensive exposition of the outcomes yielded by DifferSqueezeNet. Table (2) offers a comparative overview of our model in comparison with its baseline and other state-of-the-art architectures in the defect detection task. The table showcases performance metrics for each class as well as the average, thereby providing a holistic snapshot of our model's achievements.

DOI: 10.3844/jcssp.2025.549.557

From Table (2) we notice that the model not only presented a significant improvement compared to its baseline but also surpassed the PaDiM (Defard *et al*., 2021) model. Additionally, we can see that it presents a similar performance in AUROC to the CFLOW-AD. (Gudovskiy *et al*., 2022) model.

It is important to highlight that this novel model retains the anomaly gradient maps that were originally presented by DifferNet (Rudolph *et al*., 2021). This aspect offers valuable insights into model interpretability.

In this context, Fig. (3) showcases a collection of samples of gradient maps generated by our model, further illustrating this interpretive facet. In this figure, we can see some examples of well-detected and localized defects for some of the classes from the dataset. Bright areas in the anomaly map correspond to the detected anomalies, whereas dark areas indicate normality zones.

However, in Table (2) we could also see that our model failed to beat the baseline in some classes, namely: Carpet, Tile, Zipper, and Toothbrush. We also can observe that the classes Screw and Toothbrush are the ones where we can find the more significant difference in results between the baseline and the proposed model when analyzing the AUROC metric.

A graphical analysis was attempted to try to understand the difference in the results in both Screw and Toothbrush classes. First, we plotted a histogram of the anomaly score (Fig. 4) and ROC Curve (Fig. 5) for the Screw class. In the histogram, we can see an overlap in the anomaly scores of the normal and defective images. Therefore, some anomalies could be missed if a low threshold is selected. The ROC Curve reinforces this observation, in which we can see that we only get a 0.90 True Positive Rate (TPR) when the False Positive Rate (FPR) is at least 0.2.

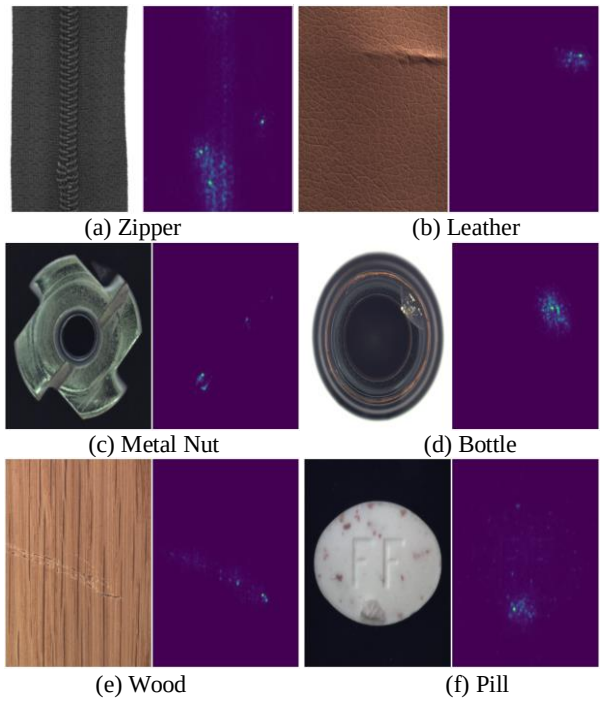


Fig. 3: Image samples from the MVTec AD Dataset (Bergmann *et al*., 2019) and its correspondent gradient maps generated by DifferSqueezeNet

Table 2: Comparison between our model versus the state-of-the-art and the baseline model (DifferNet (Rudolph *et al*., 2021)). The metric is AUROC in %. The average result of all classes of the MVTec AD dataset (Bergmann *et al*., 2019) is in the last line. The best results are in bold

Classes	PatchCore	CFlow	PaDiM	DifferNet	DifferSqueezeNet
Carpet	98.4	98.6	99.5	92.9	91.2
Grid	95.9	96.2	94.2	84	87
Leather	100	100	100	97.1	98.9
Tile	100	99.9	97.4	99.4	99
Wood	98.9	99.3	99.3	99.8	100
Bottle	100	100	99.9	99	100
Cable	99	89.3	87.8	95.9	99.1
Capsule	98.2	94.5	92.7	86.9	95.2
Hazelnut	100	100	96.4	99.3	100
Metal Nut	99.4	99.5	98.9	96.1	99.3
Pill	92.4	92.4	93.9	88.8	92
Screw	96	90.8	84.5	96.3	92.4
Toothbrush	93.3	89.7	94.2	98.6	94.2
Transistor	100	94.3	97.6	91.1	96.7
Zipper	98.2	98.4	88.2	95.1	93.6
Average	98	96.2	95	94.9	95.9

DOI: 10.3844/jcssp.2025.549.557

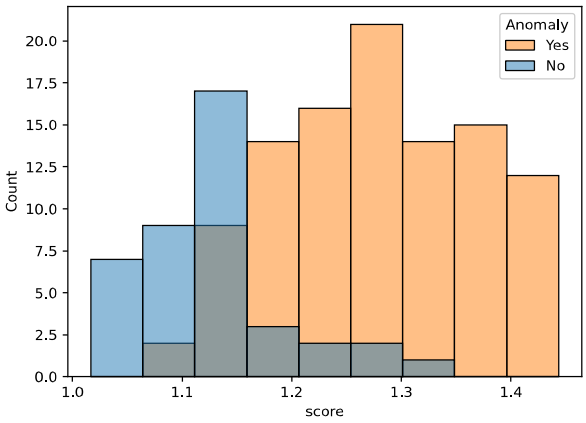


Fig. 4: Histogram of the anomaly scores generated by DifferSqueezeNet in the Screw class. We can see a relevant overlap in anomalies and normal images

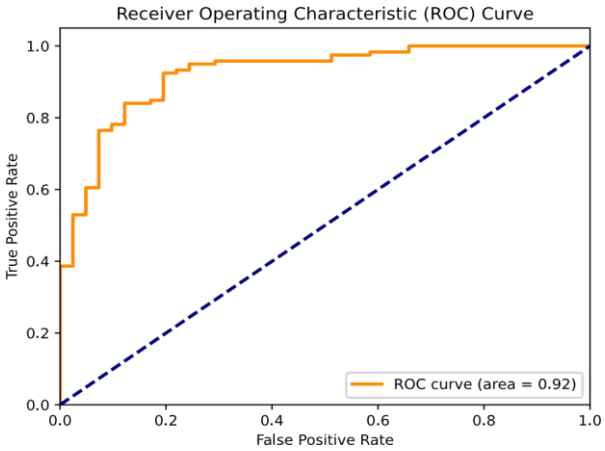


Fig. 5: ROC curve generated based on the anomaly scores predicted by DifferSqueezeNet in the Screw class

To have a contrasting point, we also generate the same visualizations for the Transistor class, on which our model surpassed its baseline. The plots are available in Figs. 6-7. In this histogram plot, a smaller overlap is presented when compared to the one presented in the Screw plot. The ROC curve demonstrates that we can achieve A TPR of 0.9 while getting a FPR of less than 0.1. This demonstrates that our model was able to learn the patterns of normal samples in this class.

Additionally, we generated gradient maps to understand the patterns that the model could be missing. In Fig. (8) we can see that our model failed to learn certain patterns in both classes Screw and Toothbrush, missing completely the anomaly in the sample images.

In the first toothbrush example, the anomaly map returned incomprehensible results, while in the second example, it flagged a part of the background as the anomaly. In the screw example, the model returned an anomaly map that marks almost all of the objects, missing completely the defect on the head of the screw in the first example and returning an inexistent defect on the head of the second screw.

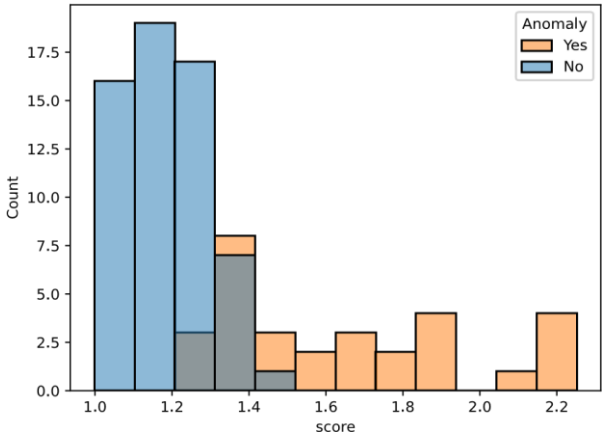


Fig. 6: Histogram of the anomaly scores generated by DifferSqueezeNet in the Transistor class. We can see that the overlap is smaller than the one in Fig. (4)

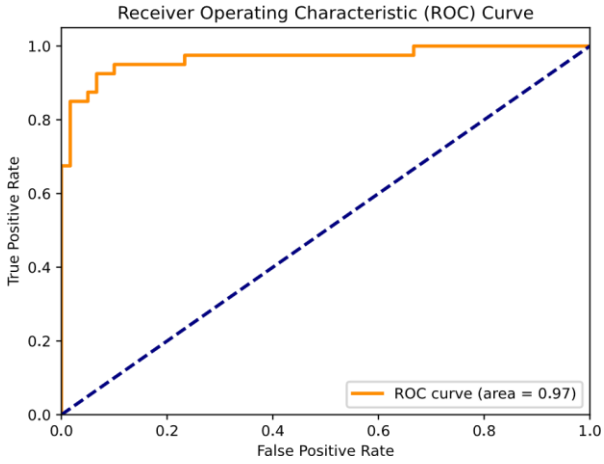
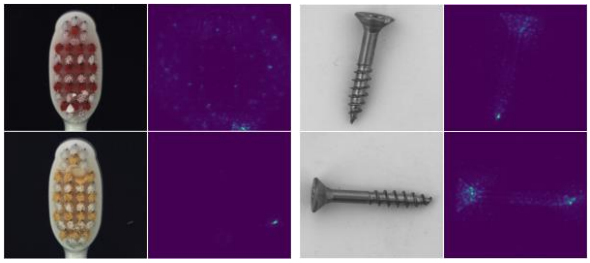


Fig. 7: ROC curve generated based on the anomaly scores predicted by DifferSqueezeNet in the Transistor class



(a) Toothbrush (b) Screw

Fig. 8: Gradient maps generated by our final model applied in image samples from the MVTec AD Dataset. The model seemed to fail to learn the patterns in both classes Screw and Toothbrush

Table 3: Comparison between our model memory requirement versus the state-of-the-art and the baseline model (DifferNet). Models trained in the MVTec AD dataset

Model	Feature extractor	Memory (GB)
PatchCore	Resnet-50	0.205
CFLOW-AD	Resnet-50	1.500
PaDiM	Resnet-50	3.800
DifferNet	AlexNet	0.240
DifferSqueezeNet	SqueezeNet	0.007

Memory Efficiency

Additionally, we observed that our approach capitalizes on the strategic integration of the SqueezeNet feature extractor, which presents a compact size of less than 5MB, a stark contrast to the original AlexNet feature extractor, weighing in at a hefty 233 MB. This substantial size reduction represents 97% less memory footprint.

The same dedication to efficiency can be seen in our entire model, with the remaining of our model architecture measuring just 600KB, as opposed to the 900KB of the initial design. This reduction in size while preserving functionality is important in contexts where computational resources are constrained.

We can see a comparison between our model memory requirement versus the state-of-the-art models and the baseline in Table (3). For fairness, all of the models were trained in the MVTec AD Dataset (Bergmann *et al*., 2019). The FEs used were the default provided by the authors. This table highlights that DifferSqueezeNet is the most compact model by a significant margin, being the only one with less than 10 MB.

Discussion

While many other defect detection models did not focus on the matter of memory consumption, this is one of the key benefits of DifferSqueezeNet, since it was designed with efficiency in mind. This makes it well-suited for contexts where computational resources are constrained, such as edge computing and real-time decision-making. This could be evaluated in a memory comparison between our model and the current state-of-the-art.

This study also highlights the importance of feature extractors in the memory complexity of a defect detection architecture. We chose the SqueezeNet pre-trained model as the feature extractor, which was expected to be effective in reducing memory consumption without significantly harming the model's performance. Even so, the memory consumption was reduced but the performance increased. This finding suggests that the SqueezeNet CNN is a promising candidate for use as a feature extractor in defect detection applications, contributing to the development of more efficient and effective defect detection models in the field.

The DifferSqueezeNet model demonstrates a refinement over the original DifferNet model both in memory efficiency and performance. The model was tested on the task of image-level defect detection and found to outperform the baseline model and other state-of-the-art architectures. The model achieved an average AUROC score of 95.9%, which is a significant improvement over the baseline model's score of 94.9%.

The new model not only surpassed its baseline in performance but also surpassed the well-known baseline PaDiM (Defard *et al*., 2021) and performed similarly to the recent CFlow-AD model (Gudovskiy *et al*., 2022), all while maintaining a smaller and more efficient architecture than both of these benchmarks.

However, the DifferSqueezeNet model also has some limitations. For instance, we can notice that the model failed to overcome the baseline model in certain categories. These errors hint that the small SqueezeNet pre-trained model might have some difficulties with complex objects. However, this could be compensated with further hyperparameter changes in these particular cases.

Overall, the DifferSqueezeNet model presents a promising approach to defect detection that prioritizes efficiency and practicality. Its smaller size and reduced memory footprint make it well-suited for real-world deployment scenarios and its focus on memory consumption sets it apart from other defect detection models. While there are limitations to the model, there are also opportunities for future work to further improve its performance.

Conclusion

In this study, we aimed to generate a model that would be leaner than its baseline– the DifferNet model. We planned to achieve this simplified architecture while still delivering competitive performance to the state-of-the-art.

After tests and analysis were completed, we found a new model architecture – DifferSqueezeNet - smaller than its predecessor and with better performance (measured by the AUROC metric) than the baseline model for the task of Image-level Defect Detection.

The achievements of this model also highlight the potential of the SqueezeNet CNN as an effective feature extractor in defect detection tasks, contributing to the development of more efficient and accurate visual inspection models.

We also discussed some future work opportunities of the new model, focusing on the categories that the model failed to overcome the baseline model. Our analysis hinted that some workarounds could be performed to level up DifferSqueezeNet's performance in those scenarios. Since our goal was to have an overall better performance than the baseline model, this would be beyond the scope of this study. Even so, this could represent an opportunity to enhance even more our presented DifferSqueezeNet model.

Additionally, some other improvements might still be made with this model architecture. The Defect Detection field continues to evolve and so does the Deep Learning domain - with new and more efficient backbones being released every year. Opportunities to experiment with alternative feature extractors and further sculpt the Normalizing Flows network - or similar architectures - are interesting prospects on the horizon. Moreover, different types of datasets and objects could benefit from the new model, verifying its adaptability to new use cases.

Acknowledgment

I would like to express my gratitude to the PPGEPS program from PUCPR for the scholarship and support in the research.

Funding Information

This research did not receive any specific grant from funding agencies in the public, commercial, or not-for-profit sectors.

Author's Contributions

Jose Joao Manrique Franco: Conducted the research, data analysis, code development, model training, and the written of the manuscript.

Marcelo Rudek: Conducted as the advisor to the research, analyzed the generated data, critically reviewed the study proposal development, and supervised the manuscript edition.

Ethics

This study is an original research effort with no ethical concerns raised.

Conflict of Interest

There are no conflicts of interest that may have influenced this research.

Data Availability Statement

In this research study, we have utilized the publicly available MVTec AD dataset which can be found at http://doi.org/10.1007/s11263-020-01400-4.

References

Akcaymir, S., Abarghouei, A. A., & Breckon, T. P. (2019). GANomaly: Semi-supervised Anomaly Detection via Adversarial Training. *Computer Vision ACCV 2018*, *11363*, 622–637. https://doi.org/10.1007/978-3-030-20893-6_39

Bergman, L., & Hoshen, Y. (2020). Classification-Based Anomaly Detection for General Data. *Computer Science*, *17*, 40–44. https://doi.org/10.48550/arXiv.2005.02359

Bergman, L., Cohen, N., & Hoshen, Y. (2020). Deep Nearest Neighbor Anomaly Detection. *Computer Science*, *18*, 51–33.

Bergmann, P., Fauser, M., Sattlegger, D., & Steger, C. (2019). MVTec AD — A Comprehensive Real-World Dataset for Unsupervised Anomaly Detection. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 9584–9592. https://doi.org/10.1109/cvpr.2019.00982

Bergstra, J., & Bengio, Y. (2012). Random Search for Hyper-Parameter Optimization. *Journal of Machine Learning Research*, *13*(2), 281–305.

Chandola, V., Banerjee, A., & Kumar, V. (2009). Anomaly detection: A survey. *ACM Computing Surveys (CSUR*, *41*(3), 1–58. https://doi.org/https://doi.org/10.1145/1541880.1541882

Cohen, N., & Hoshen, Y. (2020). Sub-Image Anomaly Detection with Deep Pyramid Correspondences. *Computer Vision and Pattern Recognition (Cs.CV); Machine Learning (Cs.LG)*, *2005.02357*. https://doi.org/https://doi.org/10.48550/arXiv.2005.02357

Defard, T., Setkov, A., Loesch, A., & Audigier, R. (2021). PaDiM: A Patch Distribution Modeling Framework for Anomaly Detection and Localization. *Pattern Recognition. ICPR International Workshops and Challenges*, *12664*, 475–489. https://doi.org/10.1007/978-3-030-68799-1_35

Dinh, L., Sohl-Dickstein, Jascha, & Bengio, S. (2016). Density Estimation Using Real Nvp. *Machine Learning*, arXiv:1605.08803. https://doi.org/https://doi.org/10.48550/arXiv.1605.08803

Fauser, M., Sattlegger, D., & Steger, C. (2019). Improving Unsupervised Defect Segmentation by Applying Structural Similarity to Autoencoders. *Proceedings of the 14th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications*, 372–380. https://doi.org/10.5220/0007364503720380

Gudovskiy, D., Ishizaka, S., & Kozuka, K. (2022). CFLOW-AD: Real-Time Unsupervised Anomaly Detection with Localization via Conditional Normalizing Flows. *2022 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, 1819–1828. https://doi.org/10.1109/wacv51458.2022.00188

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep Residual Learning for Image Recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 770–778). https://doi.org/10.1109/cvpr.2016.90

Howard, A., Sandler, M., Chen, B., Wang, W., Chen, L.-C., Tan, M., Chu, G., Vasudevan, V., Zhu, Y., Pang, R., Adam, H., & Le, Q. (2019). Searching for MobileNetV3. *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, 1314–1324. https://doi.org/10.1109/iccv.2019.00140 https://doi.org/10.48550/arXiv.2002.10445

Hu, C., & Wang, Y. (2022). An Anomaly Detection Method Based on Self-Supervised Learning with Soft Label Assignment for Defect Visual Inspection. *ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 3468–3472. https://doi.org/10.1109/icassp43922.2022.9746385

Iandola, F. N., Han, Song, Moskewicz, Matthew W, Ashraf, Khalid, Dally, W. J., & Keutzer, Kurt. (2016). Squeezenet: Alexnet-Level Accuracy with 50x Fewer Parameters and¡ 0.5 Mb Model Sizee. *Computer Vision and Pattern Recognition*, arXiv.1602.07360. https://doi.org/https://doi.org/10.48550/arXiv.1602.07360

Kakavand, M., Mustapha, N., Mustapha, A., Abdullah, M. T., & Riahi, H. (2015). Issues and Challenges in Anomaly Intrusion Detection for HTTP Web Services. *Journal of Computer Science*, *11*(11), 1041–1053. https://doi.org/10.3844/jcssp.2015.1041.1053

Krizhevsky, A. (2014). One Weird Trick for Parallelizing Convolutional Neural Networks. *Computer Science*, arXiv.1404.5997. https://doi.org/https://doi.org/10.48550/arXiv.1404.5997

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2017). ImageNet Classification with Deep Convolutional Neural Networks. *Communications of the ACM*, *60*(6), 84–90. https://doi.org/10.1145/3065386

Liu, W., Li, R., Zheng, M., Karanam, S., Wu, Z., Bhanu, B., Radke, R. J., & Camps, O. (2020). Towards Visually Explaining Variational Autoencoders. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 8639–8648. https://doi.org/10.1109/cvpr42600.2020.00867

Napoletano, P., Piccoli, F., & Schettini, R. (2018). Anomaly Detection in Nanofibrous Materials by Cnnbased Self-Similarity. Sensors, *18*(1). https://doi.org/https://doi.org/10.3390/s18010209

Napoletano, P., Piccoli, F., & Schettini, R. (2021). Semi-Supervised Anomaly Detection for Visual Quality Inspection. *Expert Systems with Applications*, *183*, 115275–115291. https://doi.org/10.1016/j.eswa.2021.115275

Roth, K., Pemula, L., Zepeda, J., Scholkopf, B., Brox, T., & Gehler, P. (2022). Towards Total Recall in Industrial Anomaly Detection. *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 14298–14308. https://doi.org/10.1109/cvpr52688.2022.01392

Rudolph, M., Wandt, B., & Rosenhahn, B. (2021). Same Same but DifferNet: Semi-Supervised Defect Detection with Normalizing Flows. *2021 IEEE Winter Conference on Applications of Computer Vision (WACV)*, 1906–1915. https://doi.org/10.1109/wacv48630.2021.00195

Sabokrou, M., Khalooei, M., Fathy, M., & Adeli, E. (2018). Adversarially Learned One-Class Classifier for Novelty Detection. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 3379–3388. https://doi.org/10.1109/cvpr.2018.00356

Siddiqui, S. A., Binder, A., Muller, E., & Kloft, M. (2018). Deep One-Class Classification. *International Conference on Machine Learning*, 4393–4402.

Venkataramanan, S., Peng, K.-C., Singh, R. V., & Mahalanobis, A. (2020). Attention Guided Anomaly Localization in Images. *Computer Vision – ECCV 2020*, *12362*, 485–503. https://doi.org/10.1007/978-3-030-58520-4_29

Winkler, C., Worrall, D., Hoogeboom, Emiel, & Welling, Max. (2019). Learning Likelihoods with Conditional Normalizing Flows. *Computer Science*, arXiv.1912.00042. https://doi.org/https://doi.org/10.48550/arXiv.1912.00042

Ye, F., Huang, C., Cao, J., Li, M., Zhang, Y., & Lu, C. (2020). Attribute Restoration Framework for Anomaly Detection. *IEEE Transactions on Multimedia*, *24*, 116–127. https://doi.org/10.1109/TMM.2020.3046884

Yi, J., & Yoon, S. (2021). Patch SVDD: Patch-Level SVDD for Anomaly Detection and Segmentation. *Springer Nature*, *12627*, 375–390. https://doi.org/10.1007/978-3-030-69544-6_23