Machine learning-based battery electrode foil inspection

Ines Müller[2,3], Wenjing Song[1,3], Sebastian Georgi[3], Timo Eckhard[3], and Alexander Korff[2]

[1] University of Applied Sciences Bremerhaven, Embedded Systems Design, Fritz-Erler-Straße 13, 27578 Bremerhaven
[2] University of Applied Sciences Lübeck, Department of Electrical Engineering and Computer Science, Mönkhofer Weg 239, 23562 Lübeck
[3] Chromasens GmbH, Max-Strohmeyer-Str. 116, 78467 Konstanz

Abstract This paper presents an analysis of various autoencoder methods for automated anomaly detection. Prototype image datasets of battery foils, used as anode (copper foil) and cathode (aluminum foil) in lithium-ion batteries, are generated using a line-scan camera system with different illumination setups. The objective is to design and evaluate unsupervised learning methods for surface inspection of the foils. Additionally, the impact of different illumination geometries on the classification performance of the implemented models and their inference times is investigated and analyzed. Another objective is to accelerate model inference by integrating a DPU-based architecture, focusing on optimizing runtime performance for real-time anomaly detection. Using the DPU, an approach achieved a speedup by a factor of 40 compared to computations on the CPU.

Keywords Autoencoder, unsupervised machine learning, anomaly detection, DPU acceleration, hardware acceleration

1 Introduction

The detection of defects in industrial production is crucial as product anomalies can lead to increased costs, delays, and quality issues. In recent years, the production of lithium-ion batteries has significantly expanded due to the rising demand for electronic devices and electric

DOI: 10.58895/ksp/1000174496-6 erschienen in:
Forum Bildverarbeitung 2024
DOI: 10.58895/ksp/1000174496 | https://www.ksp.kit.edu/site/books/m/10.58895/ksp/1000174496/

vehicles. Quality assurance plays a vital role in ensuring that the produced batteries meet performance standards. This includes the quality of the battery electrode foils, the anode, and the cathode, which are later used in batteries. Early detection of anomalies in these foils is essential to identify potential production errors or quality problems. This is where anomaly detection using machine learning methods, such as the autoencoder, comes into play. The autoencoder is a special type of neural network that can be used for unsupervised or semi-supervised anomaly detection [1]. Unsupervised methods are particularly suitable for industrial anomaly detection because labeled defect data are often scarce, expensive, or difficult to obtain.
For this reason, the following study investigates various methods for automated anomaly detection in the context of anode and cathode battery foils. To ensure a comprehensive analysis of autoencoder methods, these will be compared with methods based on similarities between data points extracted from pre-trained neural networks. Furthermore, the implemented methods will be compared with state-of-the-art approaches in industrial anomaly detection, like *Patchcore* [2] and *PaDim* [3].

Another objective includes examining the impact of different lighting conditions on the application-specific properties of the foils. For this purpose, datasets will be created under various lighting conditions, including both defect-free training data and defect anomaly data. The goal is to find a suitable lighting geometry and a method appropriate for the respective applications of the cathode and anode.

As the complexity of machine learning models increases, the demand for computational resources becomes more stringent. Traditional CPU and GPU implementations may struggle to meet the strict real-time processing requirements of industrial applications. Therefore, achieving real-time anomaly detection in industrial environments requires not only effective detection methods but also optimized inference speed to meet operational demands. To address this issue, the study also explores accelerating model inference using DPU-based hardware architectures. By deploying anomaly detection models on a DPU, inference speed and runtime performance can be significantly enhanced, enabling the real-time deployment of complex machine learning models and bridging the gap between advanced detection techniques and practical industrial applications.

2 Materials and Methods

2.1 Data Acquisition and Preprocessing

A line-scan-system (Figure 1) is used to create the datasets for anode and cathode. The foils are illuminated using different geometries: one brightfield and two darkfields. A bright field lighting technique makes reflecting surfaces appear bright since the angle at which the light is incident and the angle at which the camera is aimed are equal. Conversely, dark field illumination involves observing the light that has been dispersed or refracted from the sample. The goal is to identify different types of anomalies.

The line-scan system records image information line by line (8192 pixels per line), and the foils movement over a roller enables the assembly of these lines into a complete surface image. Each line is captured three times, with the different lighting geometries each time. This setup allows capturing the same area under varied lighting conditions, and through a line shift called deinterlacing, the images are separated into three distinct ones for further processing.

Initially, images of undamaged foils are captured to serve as the baseline for training sets. Subsequently, anomalies such as dust, scratches, and moisture are introduced to create test datasets. The influence of the three lighting setups is demonstrated in Figure 2.

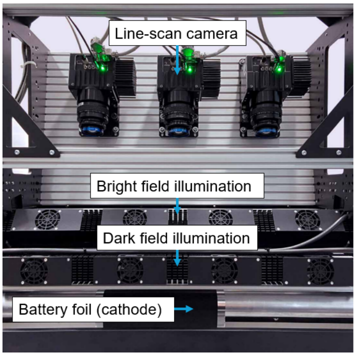
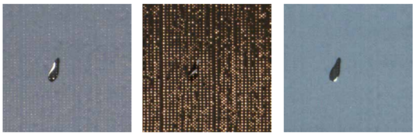
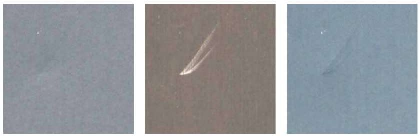


Figure 1: Line-scan-vision platform while scanning the cathode.



(a) Anode



(b) Cathode

Figure 2: Images of same sample material under different illumination geometries (dark field back, bright field, dark field front).

The preprocessing strategy is based on the assumption that different defects are visible under different lighting conditions. Images from each lighting condition are split into patches (256x256x3), transformed into grayscale (256x256x1), and combined into a *multi-flash* image (256x256x3). This combination stores relevant information from each lighting condition in separate color channels, facilitating the recognition of various anomaly types in a single image.

2.2 Solution Approach 1: Reconstruction-Based Methods

To classify the anomalies in the generated datasets, two autoencoder methods were initially tested: Convolutional Autoencoder (CAE) [4] and Variational Autoencoder (VAE) [1]. Autoencoders learn to reconstruct an image from *error-free* data that closely resembles the original. During inference, images with errors are reconstructed by the model as if they had no anomalies. By comparing the original input image with the reconstruction, such as using the *Mean Squared Error* (MSE), anomalies can be classified. For the reconstruction-based methods, *Mean Squared Error* (MSE) and *Structural Similarity Index* (SSIM) are used as classification metrics. *MSE* is widely used and quickly computed, making it suitable for high-speed applications. However, it can be sensitive to noise. *SSIM*, on the other hand, considers brightness, contrast, and structure, providing robustness against noise [5] . Both metrics help determine anomaly scores and set thresholds for binary classification based on F1-score (harmonic mean of presision and recall) maximization.

2.3 Solution Approach 2: Similarity-Based Embedding Methods

This approach involves using pre-trained neural networks (backbones) to extract features from *error-free* training data, forming embeddings that are then reduced using *Principal Component Analysis* (PCA). Classification methods such as *k-Nearest Neighbors* (kNN) and *Kernel Density Estimation* (KDE) compare the similarity of these embeddings to detect anomalies.

ResNet-50 and *MobileNet* are chosen as backbones. *ResNet-50* is suited for extracting complex features and structures in image data, making it ideal for patterned surfaces like the anode foil. *MobileNet* is selected for its efficiency and suitability for resource-constrained environments.

The extracted feature embeddings of the error-free images are stored as vectors after dimensionality reduction. During inference, the Euclidean distance to the *k*-nearest neighbors in the embeddings is calculated. The features of anomalous images are further away from the stored features of error-free images. The mean of the calculated distances then forms the anomaly score for this method.

The choice of these classifiers is motivated by the need for efficiency and the generally low complexity of the image structures involved. These approaches are based on *state-of-the-art* methods such as *Patchcore* [2], *PaDim* [3], and a method from the TKH Group (*TKH-AD*) [4], which are also compared in the evaluation. Figure 3 shows the process of the *Similarity-Based Embeddings* approach. All approaches were implemented in Python using TensorFlow and Keras.

2.4 DPU acceleration Solution

To meet the demanding processing speeds required in the battery foil industry, the Xilinx Deep Learning Processing Unit (DPU) [6] IP core was selected for hardware acceleration. The DPU, integrated into the

[4] `https://www.tkhgroup.com/`

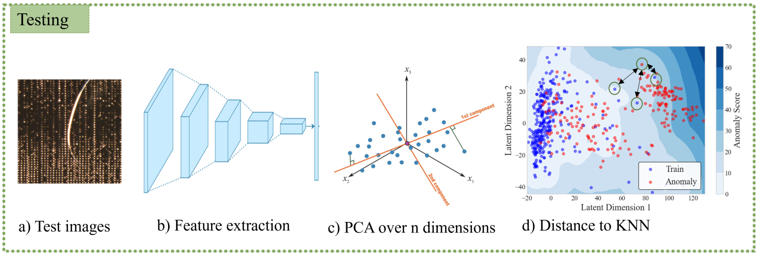


Figure 3: Testing or inference procedure of the similarity-based KNN method.

Xilinx ZCU102 FPGA platform, is designed to accelerate Convolutional Neural Network (CNN) computations using dedicated hardware optimized for parallel processing and high throughput. The architecture is configurable, supporting up to four cores [7], with a maximum of three cores used on the ZCU102 due to resource constraints. Each core independently handles deep learning tasks, maximizing resource utilization through multi-core and multi-threaded processing.

The DPU's specialized instruction set efficiently manages CNN operations such as convolutions and activation functions, making it suitable for real-time applications. Models must be quantized and compiled using Xilinx's Vitis AI tools to optimize them for the DPU, with unsupported operations offloaded to the ARM CPU. This study focused on deploying a quantized Convolutional Autoencoder (CAE) model on different DPU configurations, analyzing the impact of multi-threading on inference speed and how quantization affects the model's accuracy, using the cathode dataset.

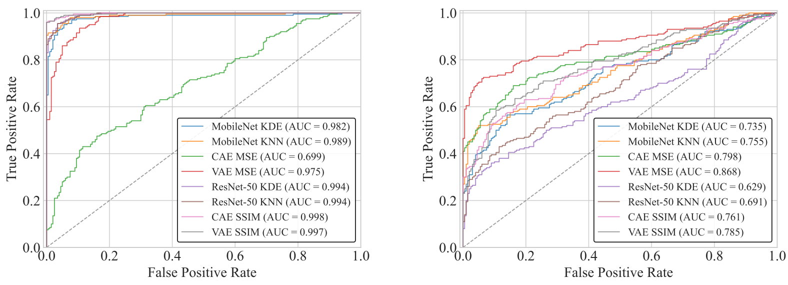
3 Results and discussion

3.1 Performance Evaluation under Different Illumination Geometries

This section presents the evaluation of model performance by analyzing their *Receiver Operating Characteristic* (ROC) curves and *Area Under the Curve* (AUC) values. The *ROC* curve illustrates the true positive rate against the false positive rate at various threshold settings, while the *AUC* value provides a single measure of overall model performance by quantifying the area under the *ROC* curve. The models were trained with 300 good samples per dataset, and the evaluation metrics were calculated on a test set with 200 good and 200 bad samples. Initially, we evaluated the implemented approaches using combined illumination geometries (*Multi-Flash*).

Figure 4 show the *ROC* curves for the cathode and anode, respectively, under *Multi-Flash* illumination. The *ROC* curves for the anode are significantly lower than those for the cathode. To examine the impact of the individual illuminations, the best models for each illumination category were tested and summarized in Figure 5. The results indicate that the combined illumination (*Multi-Flash*) does not enhance performance for the anode, with the best performance achieved using dark field back illumination alone, where MobileNet with KNN classifier reached 97% *AUC*. For the cathode, multiple approaches achieved 99% *AUC* under both *Multi-Flash* and bright field illumination.

For a comprehensive comparison with *state-of-the-art* methods, Figure 6 presents the AUC values and F1 scores for the anode data under dark field back illumination. With this Dataset the MobileNet KNN approach achieved a slightly higher AUC (97%) compared to *Patchcore* and *PaDim* (both 94%). However, *Patchcore* achieved the best performance under *Multi-Flash* illumination with 88% AUC, while *PaDim* performed best under dark field front illumination with 97% AUC for the anode data.



(a) Cathode **(b)** Anode

Figure 4: ROC-Curves with combined *Multi-Flash* images.

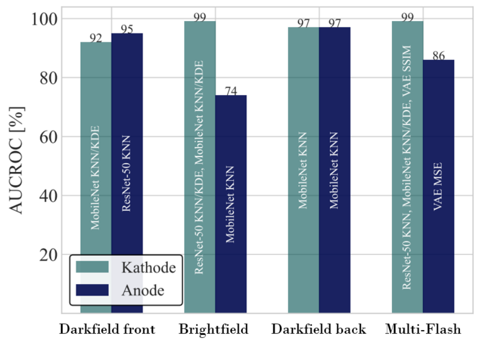


Figure 5: Comparison of the best approaches per illumination geometry.

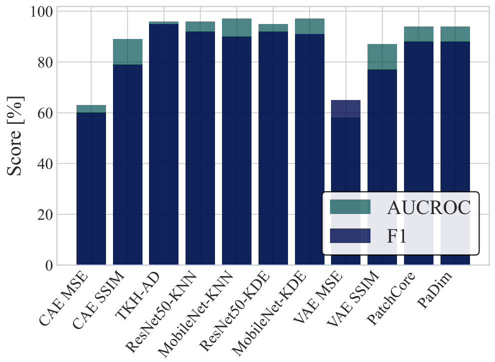


Figure 6: Comparison of the classification performance for the anode in the dark field back with *state-of-the-art* methods.

3.2 Speed evaluation

The inference speed (time for predicting, if one patch is normal or anormal) of the implemented methods is measured across the entire test dataset. The measurements were taken on a *NVIDIA GeForce RTX 3090* GPU and averaged over all test samples to relate speed to classification performance (AUC). The best time was achieved by the CAE and corresponds to a line rate of 0.076 kHz. The measurements in-

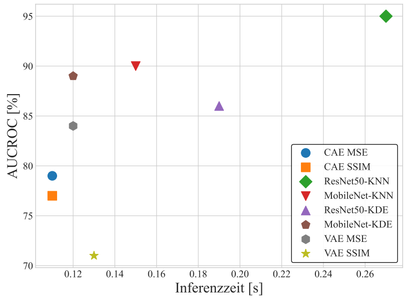


Figure 7: Inference speed of the implemented models at the anode in the darkfield front.

clude the predictions or reconstructions made by the autoencoder and the feature extraction from the pre-trained networks (calculated on the GPU), as well as the computation of the post-processing on the CPU (Intel Core i7).

3.3 Performation Evaluation of Hardware acceleration

Figure 8 is a performance comparison line chart that illustrates the time efficiency of three different configurations, labeled as *1DPU*, *2DPU*, and *3DPU*, across various thread counts while processing a single frame. In studying the impact of DPU core count and thread count on acceleration performance, it was found that performance improvements are not linear as the number of DPU cores and threads increases. When the thread count reaches a certain level, the performance of a single DPU core tends to saturate, and adding more threads may actually lead to a decline in performance. In multi-core configurations, although increasing the number of cores can enhance performance, the complexity of coordinating multiple cores and resource contention limits the extent of these improvements.

For the CAE model used in this study, the optimal configuration, identified through optimization analysis, is a combination of two DPU cores with four threads. Under this configuration, model inference achieved a line rate of 2.65 KHz (32 Patches). This result demonstrates the significant performance improvement in model inference within a

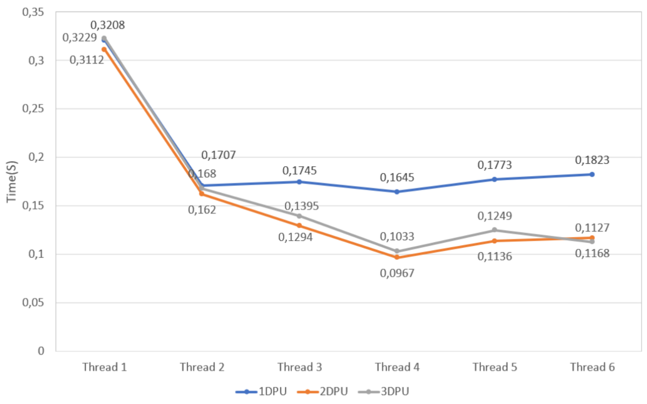


Figure 8: Thread-based performance analysis of single and multi-DPU Configurations.

DPU-accelerated environment.

Quantization was found to degrade the reconstruction accuracy of the models, particularly for MSE-based approaches due to their sensitivity to pixel-level variations, whereas SSIM-based models maintained greater robustness, demonstrating better tolerance to the effects of reduced precision.

4 Conclusions and outlook

This work developed and evaluated unsupervised machine learning methods for detecting anomalies in battery foils under various lighting conditions. Additionally, by using Xilinx's DPU IP core and Vitis AI tools for hardware acceleration, we achieved significant improvements in the model's speed and efficiency. This highlights the benefits of FPGA-based solutions for industrial applications that require fast and power-efficient performance. The investigations showed that combining different illumination geometries into a single image proved effective for the cathode, while the anode did not benefit from this approach. The best results for the anode were achieved using only the dark field back illumination. Here, the best approach (*MobileNet-KNN*) delivered slightly better results compared to established *state-of-the-art* methods such as *PatchCore* and *PaDim*. For datasets containing more structure (anode, multi-flash), *PatchCore* achieved higher results compared to MobileNet-KNN.

To obtain the most realistic results from the models, annotation by an expert would be necessary. Furthermore, saturation of *AUC* values was observed for several approaches in the cathode datasets. A future approach would be to generate and annotate datasets with even more subtle anomalies to better compare the approaches.

The findings have also demonstrated the significant improvements in processing speed and efficiency afforded by DPU acceleration, making these systems suitable for scenarios where rapid data analysis is critical.

For future advancements, focusing on further reducing latency in data processing and optimizing the entire computational pipeline will be crucial. This includes not only enhancing the model inference stages but also streamlining data input/output operations, preprocessing, and postprocessing. Real-time applications often involve continuous data streams, necessitating systems that can maintain high processing speeds without bottlenecks.

The concept of *Whole Application Acceleration*(WAA) is particularly promising. Considering the substantial improvements in processing times and efficiency achieved through DPU acceleration in this study, future research could further expand the scope of acceleration. By employing FPGA or *High-Level Synthesis* (HLS) not only for model inference but also for preprocessing and postprocessing, the entire computational pipeline, from data acquisition to final output, could benefit from hardware acceleration. Implementing WAA would lead to a more comprehensive utilization of FPGA capabilities, minimizing CPU dependencies and alleviating the bottlenecks observed in the current setup.

Acknowledgements

We acknowledge Chromasens GmbH for providing measurement equipment and technical support.

References

1. D. P. Kingma and M. Welling, “Auto-encoding variational bayes,” *arXiv preprint arXiv:1312.6114*, 2013.
2. K. Roth, L. Pemula, J. Zepeda, B. Schölkopf, T. Brox, and P. Gehler, “Towards total recall in industrial anomaly detection,” 2022.
3. T. Defard, A. Setkov, A. Loesch, and R. Audigier, “Padim: a patch distribution modeling framework for anomaly detection and localization,” 2020.
4. J. K. Chow, Z. Su, J. Wu, P. S. Tan, X. Mao, and Y. H. Wang, “Anomaly detection of defects on concrete structures with the convolutional autoencoder,” *Advanced Engineering Informatics*, vol. 45, p. 101105, 2020.
5. P. Bergmann, S. Löwe, M. Fauser, D. Sattlegger, and C. Steger, “Improving unsupervised defect segmentation by applying structural similarity to autoencoders,” pp. 372–380, 2019. [Online]. Available: http://arxiv.org/pdf/1807.02011v3
6. Xilinx, “Dpu ip details and system integration,” https://xilinx.github.io/Vitis-AI/3.0/html/docs/workflow-system-integration.html?highlight=thread, 2020, last Accessed: 2023-12-03.
7. Xilinx, “Dpuczdx8g introduction,” https://docs.xilinx.com/r/4.0-English/pg338-dpu/Introduction?tocId=Bd4R4bhnWgMYE6wUISXDLw, June 2022, last Accessed: 2024-01-10.