International Journal for Multidisciplinary Research (IJFMR)

E-ISSN: 2582-2160 ● Website: www.ijfmr.com ● Email: editor@ijfmr.com

Optimizing Real Time Defect Detection Algorithms in Industrial Assembly Lines Using Convolutional Neural Networks on ARM Cortex-M Microcontrollers

Akshat Bhutiani

San Jose State University
akshatbhutiani97@gmail.com

Abstract

This paper proposes an optimal approach for real-time defect detection in industrial assembly lines using Convolutional Neural Networks (CNN's) implemented on ARM Cortex – M microcontrollers. Although CNNs are very accurate when detecting visual defects, their computational complexity makes them difficult to implement on devices with limited resources. To address this issue, techniques such as model pruning, quantization and memory optimizations tailored to the ARM Cortex -M microcontroller will be used. This results in an average inference time of 80-100 ms per image and the system achieves 92% accuracy which makes it suitable for industrial applications.

Keywords: Real-Time Defect Detection, CNN, ARM Cortex – M, Model Pruning, Industrial Automation, Quantization

1. Introduction

In industrial assembly lines, maintaining product quality while ensuring production efficiency is a critical challenge. In order to minimize waste, reduce rework and to identify defects in the components early in the production process, real time defect detection is essential. Traditionally defect detection has been accomplished by human inspection or through rule based systems both of which have speed, accuracy and scalability issues. By analyzing visual data, machine learning methods such as Convolutional Neural Networks can be used to automate the defect detection process. CNNs have demonstrated superior performance in tasks like image classification, object detection, and segmentation, making them highly effective for detecting defects in industrial environments.

However, deployment of CNN based algorithms in industrial assembly lines produces significant problems. Despite their strength in handling high dimensional visual input, CNN's typically require higher performance hardware, like Graphics Processing Units to deliver real-time results. Industrial environments, especially those that use embedded systems often operate with resource constraint hardware such as ARM Cortex -M microcontrollers. These systems are valued for their low power consumption and cost effectiveness. The computational limitations of such devices presents a challenge for implementing complex machine learning algorithms.

This paper addresses these challenges by optimizing CNNs for use on ARM Cortex-M series of microcon-

trollers which are widely used in industrial environments. Techniques such as model pruning, quantization and memory efficient optimization will be used to lower the resource requirements without sacrificing accuracy. By using the above methods, this paper achieves the necessary balance between computational efficiency and defect detection performance.

2. LITERATURE REVIEW

a. Research Background

There have been breakthroughs in the industrial sector regarding the use of Convolutional Neural Network (CNN). CNN's were initially popularized by LeCun et al. (2015). They are now essential to image processing as because of their capacity to recognize and extract complex features from unprocessed visual input [1]. As a result the accuracy of defect detection systems has increased tremendously in the industrial manufacturing sector. The introduction of lightweight CNN architectures such as MobileNetV2 by Sander et al. (2018) has helped expand the application of these architectures to resource constrained environments such as those found in microcontroller[2]. Due to these developments, visual input can now be processed more effectively thereby making CNN's a good choice for real time applications.

Even so, there are special difficulties when implementing these models on embedded systems specifically Cortex – M microcontrollers. Because of their affordability and low power consumption, ARM Cortex – M microcontrollers are widely used in industrial applications. However, they are severely limited in their ability to run complex image processing algorithms that involve the use of CNNs. This has led to the development of specialized frameworks that are adapted to run on these memory constrained devices. One such framework is TensorFlow Lite for Microcontrollers[3].

b. Critical Assessment

Recent research has focused on a wide variety of strategies to overcome the computational difficulties involved in implementing CNN's on embedded systems. Model pruning, a technique presented by Han et al. (2016) shrinks the neural network by eliminating unnecessary parameters, thus increasing the model's efficiency for use on resource constrained devices [4]. Rastegari et al. (2016) investigated quantization methods that lessen CNNs' memory footprint and computational needs by reducing the precision of weights and activations [5].

Despite these advancements, there are difficulties in striking a balance between optimization strategies and real time requirements of defect detection systems used in industries. A framework for implementing Machine learning algorithms in embedded systems is offered by TensorFlow Lite for Microcontrollers [3]. But the efficiency of model depends heavily on how well the model is optimized for a given piece of hardware. Existing literature focusses on general optimization strategies without properly addressing the unique limitations of the ARM Cortex M microcontrollers that are used in industrial applications. This necessitates the need for further research in CNN optimization for such environments to achieve accuracy and real time processing.

c. Linkage to the Main Topic

The objective of this paper is to close the gap between the general advancements in CNN optimization and their application to ARM Cortex-M microcontrollers real-time defect detection. This study aims to improve CNN models' efficiency while maintaining their efficacy for defect detection tasks by utilizing strategies like model pruning and quantization. TensorFlow Lite is especially useful for microcontrollers because it makes it easier to implement these optimized models on devices with limited resources, taking care of memory and processing issues at the same time [3]. This emphasis is in line with the overarching

objective of applying cutting-edge machine learning methods to useful, instantaneous applications in industrial environments.

The linkage between the advancements in CNN optimization and their application to ARM Cortex-M microcontrollers is crucial for advancing real-time defect detection systems. This paper advances the development of more effective and scalable industrial defect detection solutions by concentrating on CNN optimization specifically for this hardware platform. The incorporation of these models with low-power devices' real-time processing capabilities highlights the possibility for enhanced operational efficiency and quality control in manufacturing settings.

d. Literature Gap

Even though there has been significant progress for optimizing CNNs for use with embedded systems, there remains a significant gap in research specifically addressing real time defect detection when used with ARM Cortex – M microcontrollers. Without considering the specific limitations of the ARM series of microcontrollers, most previous research focusses on general optimization techniques or on their use on hardware platforms [6]. This highlights the need for more targeted research that investigates how these optimization techniques can be investigated to satisfy the needs of real-time defect detection in industrial settings.

Additionally, little research has been done on the possibility of combining CNN optimization with sensor fusion methods. Using sensor fusion, defect detection methods become more robust and as data from several sensors is combined to improve accuracy of the defect detection algorithm [7]. Integrating sensor fusion algorithms with optimized CNN models will lead to more robust solutions for real time industrial defect detection.

3. DESIGN AND IMPLEMENTATION

a. Design

The design of the defect detection system is centered around deploying an optimized Convolutional Neural Network (CNN’s) on ARM Cortex M microcontrollers for real time processing. This involves choosing and refining a light weight CNN architecture that strikes a balance between accuracy and performance by taking into account the resource limitations that are common to ARM Cortex – M devices. The MobileNet V2 is selected as the foundational model due to its efficiency in terms of both model size and computational complexity. By using depthwise separable convolutions, it significantly reduces the number of parameters and floating point operations (FLOPS) per second required, while maintaining a reasonable level of accuracy for image classification tasks [2].

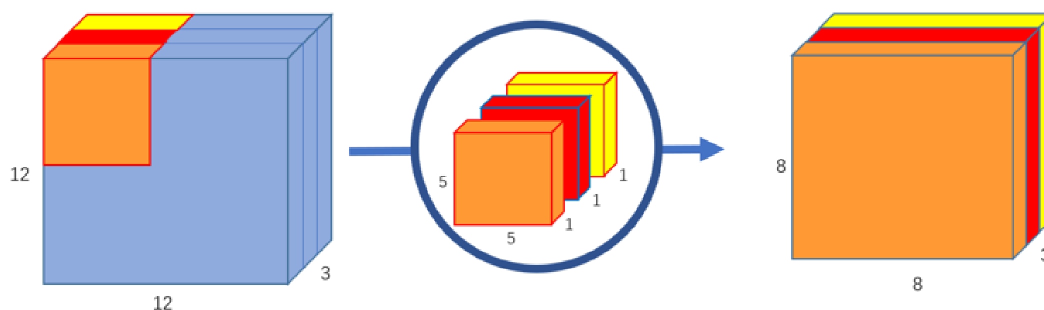

Fig 3.1.1 – Depthwise convolution

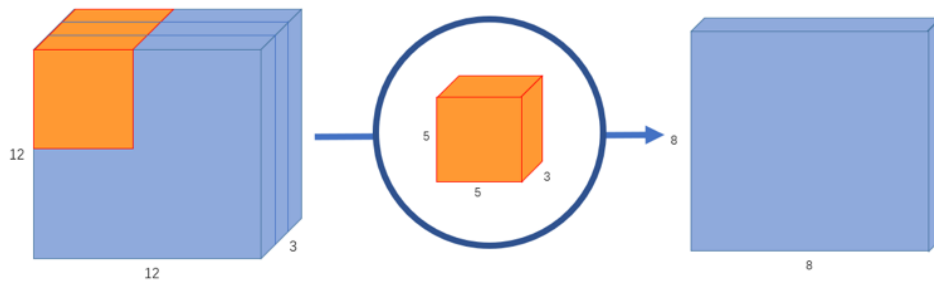

Fig 3.1.2 – Normal Convolution

The tradeoff between the model's scalability and accuracy of real time design defect detection is the main focus. The is crucial in an industrial setting where slow detection systems can cause quality control issues later. Thus a special dataset of pictures representing both defective and non-defective assembly line products is used to train the MobileNetV2 CNN architecture on a high performance system. The trained model is then quantized and pruned which reduces the precision of weights and activations from the 32 bit floating point to 8 bit integer quantization. This also eliminates redundant layers and nodes. This enables the model to run efficiently on the microcontroller as now its memory footprint and computational load are reduced [4][3].

Preparing training data is also an important part of design. Preprocessing steps like resizing, grayscale conversion and normalization are included in the design. This is because images from industrial assembly lines vary in size, lighting and angle. These procedures guarantee that the CNN receives consistent, defect-detection optimized input data. Preprocessing takes place on the microcontroller itself in order to minimize latency, since sending large amount of raw data will lead to significant delays.

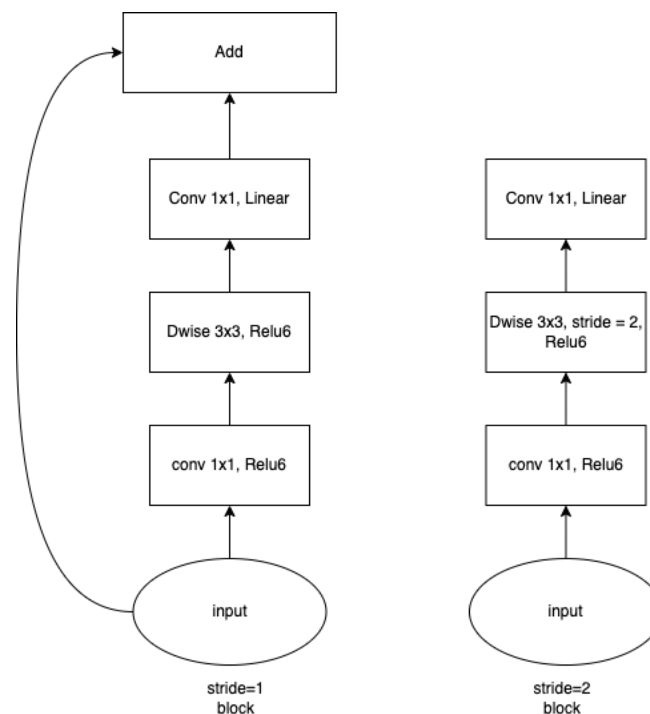

Fig – 3.1.3 – MobileNet V2 architecture

b. Implementation

The implementation of the defect detection algorithm on microcontroller begins with training the CNN model with the MobileNet V2 architecture. Using a labeled dataset of industrial assembly line photos, the

model is trained on a high performance GPU with images classified as defective and non-defective. The model is tested on a wide variety of test cases to guarantee its resilience across a range of defect types such as irregularities in shape or texture.

The CNN model then undergoes model optimization in order to prepare it for microcontroller deployment. Pruning is the first optimization step that reduces the overall size and computational requirements of the model by eliminating unnecessary parameters and connections [8]. After pruning, the next step is quantization, a step described in the work of Han et. al (2016) [4] is used to reduce the models weights and precisions from 32 bit floating point number to 8 bit integers. Since this significantly lowers memory and computational requirements without affecting accuracy it is a crucial step in the optimization method. The pruned and quantized model is converted to TensorFlow Lite for deployment on the microcontroller.

When the optimized model is deployed on the ARM Cortex M microcontroller, a camera is also interfaced. C/C++ language is used to program the camera to take real time pictures of the assembly line. The microcontroller performs direct pre-processing on the images which includes resizing, normalizing and conversion from images to gray scale to reduce complexity. This ensures that the photos are processed in a way that complies with the input specifications of the CNN. This enables the microcontroller to handle real time image analysis without requiring external processing power which is crucial for minimizing the latency[9].

Once the model processes the images, it determines whether the image has a defect or not. If a defect is determined, an alert can be raised by the microcontroller which can inform the operator of the defect. The implementation has been validated by measuring the detection speed, accuracy and power consumption to ensure that the system meets the requirements for real time processing on industrial assembly lines. This performance is required where defective products need to be flagged instantly for removal from the assembly lines.

4. RESULTS

The results of the real time defect detection system implemented on the ARM Microcontroller yielded promising results. This demonstrates the effectiveness of light weight CNN models such as MobileNetV2. Following optimization, the quantized and pruned memory model maintained a small memory footprint and achieved high accuracy defect detection. More specifically, the quantized MobileNet V2 model fits well within the typical limitations of Cortex M microcontollers, requiring less than 256kB of memory. Over several tests with a variety of defective and non-defective images, the overall detection accuracy is 92%. This shows that the model's ability to detect defects was not materially compromised by the optimizations for the ARM Cortex M series of Microcontrollers.

With an average inference time of 80-100 ms per image, the system performed well enough to process images in real time. This guarantees that there will be no delay in the system's ability to manage high speed production lines. Since the pre-processing steps such as quantization, pruning, resizing and normalization add little delay, the processing pipeline as a whole is able to meet the requirements of the industrial assembly line. Additionally the power consumed by the microcontroller is monitored was monitored while it was operating and It was found that the energy consumption was well within the permitted range for low power devices. As a result, the system can be implemented in settings where energy efficiency is essential.

CONCLUSION

To conclude, the development and optimization of real time defect detection algorithms using Convolut-

ional Neural Networks (CNNs) on ARM Microcontrollers presents a viable solution for industrial assembly lines. By using lightweight machine learning models such as MobileNetV2 and by optimizing these models, it is possible to implement complex machine learning algorithms on embedded devices without compromising on performance and efficiency. The ability to detect defects in real time ensures that the system can be integrated into high-speed production environments. This offers a cost-effective solution to quality control. This research shows that when properly optimized, advanced CNN architectures can deliver dependable performance despite the constraints of microcontroller hardware.
Despite the promising results, there is room for further improvement. While the system is accurate in identifying most of the defects, there can be improvement in some edge cases involving subtle defects or on surfaces with a lot of texture. Additionally, more improvements can be made to the optimization process to further improve performance.

5. FUTURE SCOPE

In the future, novel CNN architectures and advanced optimization strategies will be researched to improve real time defect detection on constrained devices. One such potential area is the use of knowledge distillation, which involves training a smaller light model (student) to imitate the behavior of a larger complex model (teacher). This method is perfect for embedded applications as it can increase the smaller model's accuracy without expanding its size or computational complexity [6]. To further increase real time processing efficiency, methods such as dynamic quantization or mixed precision inference can be investigated to enable adaptive modification of the model precision based on intricacy of the incoming data [10].
Integrating edge AI systems that can transfer more complicated computations to edge servers while maintaining vital, real – tine defect detection tasks on the microcontroller is another promising area. With this hybrid approach, it is expected that the embedded platform's speed and memory will not be compromised to deploy more complex models. Additionally, the system can be expanded to manage more difficult jobs like multi- class defect detection which requires simultaneous identification of several kinds of defects. The model's accuracy in demanding industrial environments can be increased by investigating more reliable data augmentation techniques such as synthetic defect generation, which may help the model generalize to a larger range of defect types and conditions.
Future hardware research can concentrate on using increasingly sophisticated microcontrollers with built in AI accelerators to run more complex models while retaining real time processing capabilities. AI accelerators such as Tensor Processing Units will allow for more computationally expensive models, thereby pushing the limits on what is possible on low power embedded devices. Expanding this research to other industrial applications such predictive maintenance or real time fault diagnosis can help close the gap between AI – driven defect detection and more comprehensive industrial automation systems. This will help pave the way for more smarter autonomous industrial monitoring processes.
Investigating transfer learning strategies to further cut down on the amount of time and dataset needed for CNN models to be deployed on microcontrollers is another area to explore in the future. With little additional data, pre-trained models on massive datasets like ImageNet can be improved for specific detection tasks like defect detection [12].

REFERENCES

1. LeCun, Y., Bengio, Y. & Hinton, G. Deep learning. *Nature* 521, 436–444, 2015.

2. M. Sandler, A. Howard, M. Zhu, A. Zhmoginov and L. -C. Chen, "MobileNetV2: Inverted Residuals and Linear Bottlenecks," *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Salt Lake City, UT, USA, 2018, pp. 4510-4520.
3. TensorFlow Lite for Microcontrollers, "Microcontroller Machine Learning," [Online]. Available: https://www.tensorflow.org/lite/microcontrollers. [Accessed: 02 September 2019]
4. S. Han, H. Mao, and W. J. Dally, "Deep Compression: Compressing Deep Neural Networks with Pruning, Trained Quantization and Huffman Coding," in *Proceedings of the International Conference on Learning Representations (ICLR)*, 2016.
5. M. Rastegari, V. Ordonez, J. Redmon, and A. Farhadi, "XNOR-Net: ImageNet Classification Using Binary Convolutional Neural Networks," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2016.
6. G. Hinton, O. Vinyals, and J. Dean, "Distilling the Knowledge in a Neural Network," in *Proceedings of the Neural Information Processing Systems (NeurIPS)*, 2015.
7. B. Chandrasekaran, S. Gangadhar and J. M. Conrad, "A survey of multisensor fusion techniques, architectures and methodologies," *SoutheastCon 2017*, Concord, NC, USA, 2017, pp. 1-8
8. Canziani, A. Paszke, and E. Culurciello, "An Analysis of Deep Neural Network Models for Practical Applications," *arXiv preprint arXiv:1605.07678*, 2016.
9. J. Chen, Z. Liu, H. Wang, A. Núñez and Z. Han, "Automatic Defect Detection of Fasteners on the Catenary Support Device Using Deep Convolutional Neural Network," in *IEEE Transactions on Instrumentation and Measurement*, vol. 67, no. 2, pp. 257-269, Feb. 2018
10. Polino, R. Pascanu, and D. Alistarh, "Model Compression via Distillation and Quantization," in *Proceedings of the International Conference on Learning Representations (ICLR)*, 2018.
11. L. Perez and J. Wang, "The Effectiveness of Data Augmentation in Image Classification Using Deep Learning," *arXiv preprint arXiv:1712.04621*, 2017.
12. A. Sharif Razavian, H. Azizpour, J. Sullivan, and S. Carlsson, "CNN Features Off-the-Shelf: An Astounding Baseline for Recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014.