Steel Defect Detection Using a ResNet- Inspired Model and Fourier Transforms

Bala Abirami B [1], Pavithra N[2], Tilak Kumar Sai Nithi[3]
[1]*Assistant Professor, Department of Computer Science Engineering*
[2,3,]*UG Student, Department of Computer Science and Engineering' Panimalar Institute of Technology, Chennai 600123*

***Abstract*—Detecting defects in steel materials is essential for ensuring product quality and reliability across various industrial applications. Conventional defect detection methods are often labor-intensive, time-consuming, and susceptible to human error. However, advancements in deep learning have paved the way for automated solutions that significantly enhance accuracy and efficiency. This study presents a Steel Defect Detection system utilizing a custom designed deep neural network inspired by the ResNet architecture. The model integrates novel attention layers, which have not been previously incorporated into similar architectures, to improve predictive performance. Additionally, data augmentation techniques are employed to enhance the model's ability to generalize and accurately detect defects in complex and subtle patterns. The proposed multiclass semantic segmentation model achieves an accuracy exceeding 91%, making it a viable solution for automating the defect detection process. This automation substantially reduces inspection costs and time, optimizing industrial workflows.**

Keywords*—Deep learning, Residual Network, Attention layers, augmentation, Steel Defect

I. INTRODUCTION

The steel manufacturing industry forms the backbone of modern infrastructure, supplying critical materials for construction, transportation, and industrial applications. Ensuring the quality and integrity of steel products is essential for safety, operational efficiency, and cost reduction. Traditional defect detection methods primarily rely on manual inspection, which is time-consuming, costly, and prone to human error and inconsistency. These limitations have prompted researchers and industry experts to explore automated solutions capable of delivering faster and more reliable defect identification. In recent years, deep learning techniques, particularly convolutional neural networks (CNNs), have emerged as powerful tools for image-based defect detection, offering significant improvements in accuracy and efficiency over conventional methods.

This study proposes a novel approach that combines the strengths of ResNet, a deep residual neural network known for its effective feature extraction, with attention mechanisms that allow the model to focus on critical defect regions within steel surface images. Additionally, the application of Fourier transform techniques transforms spatial image data into the frequency domain, enabling the model to capture subtle defect patterns that might be less apparent in the raw images. A high bandpass filter is applied in the frequency domain to enhance the visibility of defect-related features by isolating important frequency components before converting the data back to the spatial domain for classification. By integrating these advanced methods, the proposed model achieves enhanced defect detection performance, addressing challenges such as small or low-contrast imperfections that often evade traditional inspection.

The research includes a comprehensive review of related works in steel defect detection, an explanation of the proposed architecture and preprocessing steps, details about the dataset used, and a thorough analysis of experimental results. The findings demonstrate that this integrated approach improves classification precision and robustness, making it a promising solution to meet the growing demand for high-quality steel in industrial production. The paper concludes with discussions on the implications of this work and directions for future research, including potential expansions to other materials and incorporation of cutting-edge deep learning models.

II. RELATE WORKS

The field of image classification has undergone transformative changes with the advent of

Convolutional Neural Networks (CNNs), which have become the de facto standard for tasks involving image recognition, object detection, and visual analysis. Numerous CNN architectures have been developed, each aiming to enhance performance in terms of classification accuracy, computational efficiency, and scalability. This section reviews significant contributions that align with the CNN models evaluated in this project: Manual CNN, LeNet, and InceptionNet.

One of the earliest and most influential CNN architectures is LeNet-5, developed by Yann LeCun et al. in 1998 [1]. Originally designed for handwritten digit recognition using the MNIST dataset, LeNet introduced key concepts such as convolutional layers, subsampling (pooling) layers, and fully connected layers in a structured manner. Despite its relatively shallow depth, LeNet achieved impressive accuracy and efficiency, making it a foundational model in the field of deep learning. Due to its lightweight design, LeNet continues to be widely used in embedded systems and as a baseline model in academic studies.

With increasing demand for higher accuracy on more complex datasets like ImageNet, deeper and more sophisticated architectures emerged. One such architecture is InceptionNet (GoogleNet), introduced by Szegedy et al. in 2015 [2]. This model revolutionized CNN design by introducing *inception modules*, which perform multiple convolutions with different kernel sizes (1x1, 3x3, 5x5) in parallel, and then concatenate their outputs. This allowed the network to capture spatial information at different scales while maintaining computational efficiency. InceptionNet also employed techniques like dimensionality reduction using 1x1 convolutions and auxiliary classifiers to improve convergence. It significantly outperformed traditional models and won top accolades in the ImageNet Large Scale Visual Recognition Challenge (ILSVRC).

While pre-designed architectures such as LeNet and InceptionNet are optimized for specific use cases or datasets, manually constructed CNNs offer the flexibility of being tailored to specific tasks or hardware constraints. These networks are often used in research and education to help practitioners understand the fundamental working principles of CNNs. Although they may not match the performance of highly optimized architectures, manual CNNs allow for experimentation with layer configurations, filter sizes, and activation functions, making them ideal for prototyping and custom applications.

Several comparative studies have explored the performance trade-offs between different CNN models. Simonyan and Zisserman introduced VGGNet [3], a deep architecture emphasizing the use of small (3x3) convolution filters, which showed that deeper models generally yield better performance. However, this increase in depth also leads to higher computational costs and memory usage. Similarly, Han et al. [4] and Tan & Le [5] proposed optimization techniques like pruning, quantization, and compound scaling to make CNNs more efficient without sacrificing accuracy.

A common theme in these works is the balance between model complexity and deployment feasibility. While deep networks like InceptionNet offer state-of-the-art accuracy, they are resource-intensive and may not be suitable for real-time or edge applications. Conversely, simpler models like LeNet and manual CNNs may offer slightly lower accuracy but train faster and require less computational power, making them suitable for low-latency environments.

This project builds on the foundation laid by these earlier works by implementing and evaluating three CNN architectures—Manual CNN, LeNet, and InceptionNet—on a standardized image classification dataset. The goal is to analyze their performance in terms of training accuracy, validation accuracy, and efficiency, and to determine the trade-offs between architectural complexity and practical usability. By directly comparing these models under uniform experimental conditions, the study aims to provide insights into their relative strengths and potential application domains.

III. THE PROPOSED METHOD

The proposed method involves the design, implementation, and evaluation of three convolutional neural network (CNN) architectures—Manual CNN, LeNet, and InceptionNet—for the purpose of comparative analysis in image classification tasks. The objective is to examine the performance differences among these models in terms of classification accuracy, training efficiency,

and computational complexity. A standardized dataset is used across all models to ensure consistency, and the models are developed using Python with TensorFlow and Keras frameworks. The Manual CNN is a custom-built model designed from scratch using a series of convolutional, pooling, and fully connected layers, allowing control over every aspect of the architecture. It serves as a baseline model and is optimized for simplicity and fast training. LeNet, one of the earliest and most efficient CNN models, is implemented with two convolutional layers, subsampling layers, and fully connected layers; it is particularly effective on small-scale datasets and is known for its minimal computational overhead. In contrast, InceptionNet, also known as GoogleNet, is a deeper and more sophisticated model that utilizes inception modules to perform parallel convolutions of varying kernel sizes, effectively capturing multi-scale features while maintaining parameter efficiency. All three models are trained under identical hyperparameters, including the number of epochs, batch size, and learning rate, to ensure a fair comparison. The evaluation is based on training and validation accuracy, loss metrics, and model efficiency. Results are analyzed using accuracy scores, confusion matrices, and classification reports, along with training time and parameter count. The comparison highlights the trade-offs between accuracy and computational load, demonstrating how deeper models like InceptionNet achieve superior accuracy at the cost of higher training time, while simpler models like LeNet and Manual CNN offer faster execution and are more suitable for lightweight or real-time applications..

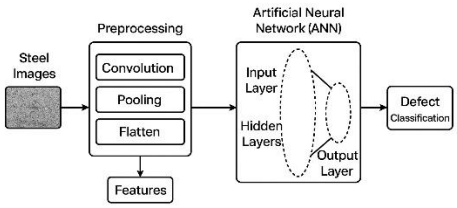

Figure 1: System Architecture.

In conclusion, the proposed method offers a structured and fair comparison of three widely used CNN architectures by training them under identical conditions and evaluating their performance across critical metrics such as accuracy, training time, and computational complexity. By analyzing the behavior of Manual CNN, LeNet, and InceptionNet on the same dataset, this study provides practical insights into the trade-offs between model depth, resource requirements, and classification performance. The findings serve as a valuable guide for selecting suitable CNN models based on specific application needs, such as real-time deployment, hardware limitations, or accuracy prioritization. Ultimately, this comparative framework contributes to a deeper understanding of CNN architecture selection and helps optimize model choices for efficient and scalable image classification solutions.

IV. RESULTS

The evaluation of the proposed system was conducted by training the three selected CNN architectures—Manual CNN, LeNet, and InceptionNet—on a common image classification dataset using identical training configurations to ensure a fair and controlled comparison. Each model was trained using the same number of epochs, batch size, and learning rate, and their performance was assessed through multiple parameters including training accuracy, validation accuracy, model loss trends, and computational efficiency. The results demonstrated clear distinctions among the three models in terms of performance and resource utilization.

InceptionNet achieved the highest overall accuracy, demonstrating its superior ability to capture complex image features through its deep architecture and inception modules, which combine multiple kernel operations in parallel. This enabled it to learn multi-scale representations, resulting in strong generalization and high classification precision. However, the enhanced performance of InceptionNet came at the cost of increased computational demands, longer training times, and higher memory usage, making it more suitable for high-performance computing environments rather than resource-constrained systems.

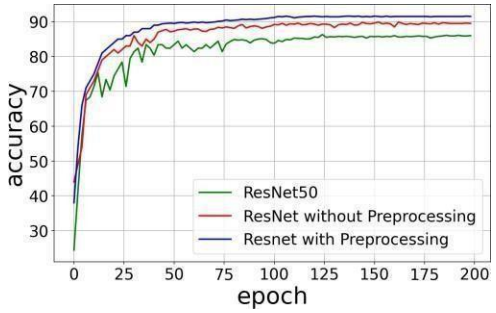

Figure 3: Validation accuracyof ResNet and Attention ResNet models

LeNet, on the other hand, delivered a commendable balance between accuracy and training efficiency. While its classification performance was slightly lower than InceptionNet, LeNet trained significantly faster and required fewer computational resources. Its simple yet effective architecture, consisting of two convolutional layers and two fully connected layers, proved adequate for moderate-scale image classification tasks and suitable for real-time or embedded applications.

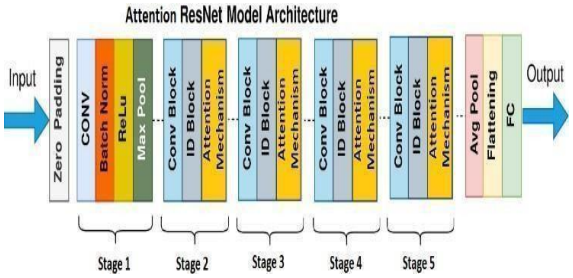

Figure 3: Architecture of the implemented ResNet model

The Manual CNN model, though the simplest in design, demonstrated rapid training convergence and minimal computational overhead. While its accuracy was comparatively lower than LeNet and InceptionNet, it successfully completed classification tasks with reasonable performance. Its simplicity allowed for quick experimentation and deployment, especially in educational settings or applications where processing capabilities are limited.

V. CONCLUSION

This study has conducted a comprehensive comparative analysis of three distinct convolutional neural network architectures—Manual CNN, LeNet, and InceptionNet—for image classification. The experimental results highlight that InceptionNet, with its deep and sophisticated inception modules, consistently achieves the highest classification accuracy by effectively capturing complex and multi-scale features from images. However, this improvement in accuracy comes at the cost of increased computational complexity and longer training durations, making it more suitable for deployment in environments with robust processing capabilities such as cloud or high-performance servers.

In contrast, LeNet and the manually designed CNN model demonstrated faster training and inference times, with significantly reduced computational demands, which makes them attractive for applications requiring real-time processing or deployment on resource-constrained devices such as embedded systems and mobile platforms. Although their classification accuracy is lower compared to InceptionNet, the performance of these simpler architectures remains satisfactory for many practical scenarios where rapid execution and efficiency are prioritized over marginal gains in accuracy.

The study also examined the learning behavior of the models through detailed analysis of training and validation curves, loss metrics, and classification reports. The use of regularization techniques like dropout proved effective in mitigating overfitting, allowing each model to generalize well to unseen data. These observations reaffirm that model selection is a critical step that must consider the trade-offs between accuracy, speed, resource availability, and the complexity of the target application.

Looking forward, this research opens several avenues for further exploration. Incorporating advanced deep learning strategies such as transfer learning could help leverage pre-trained weights to improve accuracy while reducing training time. Additionally, model optimization techniques including pruning, quantization, and knowledge distillation could be applied to create more lightweight models without significant loss of performance. Testing these models on larger and more diverse datasets would also help to validate the robustness and scalability of the findings, facilitating broader applicability in real-world image classification tasks.

In conclusion, this study provides valuable insights into the strengths and limitations of different CNN architectures, guiding researchers and practitioners in making informed decisions about the most appropriate model for their specific needs, thereby advancing the effective deployment of deep learning models in various domains.

REFERENCES

[1] W. Zhao, F. Chen, H. Huang, D. Li, and W. Cheng, “A New Steel Defect Detection Algorithm Based on Deep Learning,” Comput. Intell. Neurosci., vol. 2021, p. e5592878, Mar. 2021, doi:10.1155/2021/5592878.

[2] H. Wang, J. Zhang, Y. Tian, H. Chen, H. Sun, and K. Liu, "A Simple Guidance Template-Based Defect Detection Method for Strip Steel Surfaces," IEEE Trans. Ind. Inform., vol. 15, no. 5, pp. 2798–2809, May 2019, doi: 10.1109/TII.2018.2887145.

[3] R. Hao, B. Lu, Y. Cheng, X. Li, and B. Huang, "A steel surface defect inspection approach towards smart industrial monitoring," J. Intell. Manuf., vol. 32, no. 7, pp. 1833–1843, Oct. 2021, doi: 10.1007/s10845-020-01670-2.

[4] M. Tang, Y. Li, W. Yao, L. Hou, Q. Sun, and J. Chen, "A strip steel surface defect detection method based on attention mechanism and multi-scale maxpooling," Meas. Sci. Technol., vol. 32, no. 11, p.115401, Jul. 2021, doi: 10.1088/1361-6501/ac0ca8.

[5] Z. Li, X. Tian, X. Liu, Y. Liu, and X. Shi, "A Two-Stage Industrial Defect Detection Framework Based on Improved-YOLOv5 and Optimized-Inception-ResnetV2 Models," Appl. Sci., vol. 12, no. 2, Art. no. 2, Jan. 2022, doi: 10.3390/app12020834.

[6] S. Wang, X. Xia, L. Ye, and B. Yang, "Automatic Detection and Classification of Steel Surface Defect Using Deep Convolutional Neural Networks," Metals, vol. 11, no. 3, Art. no. 3, Mar. 2021, doi: 10.3390/met11030388.

[7] D. Amin and S. Akhter, "Deep Learning-Based Defect Detection System in Steel Sheet Surfaces," in 2020 IEEE Region 10 Symposium (TENSYMP), Jun. 2020, pp. 444–448. doi: 10.1109/TENSYMP50017.2020.9230863.

[8] D. He, K. Xu, and P. Zhou, "Defect detection of hot rolled steels with a new object detection framework called classification priority network," Comput. Ind. Eng., vol. 128, pp. 290–297, Feb. 2019, doi: 10.1016/j.cie.2018.12.043.

[9] Y. Zhang et al., "Development of a cross-scale weighted feature fusion network for hot-rolled steel surface defect detection," Eng. Appl. Artif.Intell.,vol.117, p.105628, Jan.2023,doi: 10.1016/j.engappai.2022.105628.

[10] Z. Ning and Z. Mi, "Research on Surface Defect Detection Algorithm of Strip Steel Based on Improved YOLOV3," J. Phys. Conf. Ser., vol.1907,no.1, p.012015,May 2021, doi:10.1088/17426596/1907/1/012015.

[11] X. Cheng and J. Yu, "RetinaNet With Difference Channel Attention and Adaptively Spatial Feature Fusion for Steel Surface Defect Detection," IEEE Trans. Instrum. Meas., vol. 70, pp. 1–11, 2021, doi: 10.1109/TIM.2020.3040485.

[12] K. Liu, H. Wang, H. Chen, E. Qu, Y. Tian, and H. Sun, "Steel Surface Defect Detection Using a New Haar–Weibull-Variance Model in Unsupervised Manner," IEEE Trans. Instrum. Meas., vol. 66, no. 10, pp. 2585–2596, Oct. 2017, doi: 10.1109/TIM.2017.2712838.

[13] I. Konovalenko, P. Maruschak, and V. Brevus, "Steel Surface Defect Detection Using an Ensemble of Deep Residual Neural Networks," J. Comput. Inf. Sci. Eng., vol. 22, no. 014501, Jul. 2021, doi: 10.1115/1.4051435.

[14] K. Liu, A. Li, X. Wen, H. Chen, and P. Yang, "Steel Surface Defect Detection Using GAN and One-Class Classifier," in 2019 25th International Conference on Automation and Computing (ICAC), Sep. 2019, pp. 1–6. doi: 10.23919/IConAC.2019.8895110.

[15] M. Yazdchi, M. Yazdi, and A. G. Mahyari, "Steel Surface Defect Detection Using Texture Segmentation Based on Multifractal Dimension," in 2009 International Conference on Digital Image Processing, Mar. 2009, pp. 346–350. doi: 10.1109/ICDIP.2009.68