Frontiers in Computing and Intelligent Systems
ISSN: 2832-6024 | Vol. 17, No. 1, 2026

A Systematic Benchmark of Attention Mechanisms on the NEU-DET Strip Surface Defect Dataset Using YOLOv11n

Liangliang Dai, Bowen Wang and Bin Gong *

College of Electrical Engineering, Northwest Minzu University, Lanzhou, Gansu, China
*** Corresponding author:** Bin Gong (Email: gong_bin01@163.com)

Abstract: Strip surface defect detection is an important part of the steel industry. Attention mechanisms have been widely used to enhance the feature representation capability of detection networks; however, their effectiveness on small industrial defect datasets lacks systematic evaluation. In this paper, we take the lightweight detector YOLOv11n as the baseline and integrate 15 attention mechanisms into a unified framework, covering channel attentions (SE and ECA), spatial attention (SimAM), hybrid attentions (CBAM and GAM), coordinate attention (CA), efficient multi-scale attention (EMA), and others. Strictly controlled experiments are conducted on the NEU-DET strip surface defect dataset. All models are trained under identical conditions: 50 epochs, 200×200 input resolution, batch size 16, 70/30 train/validation split, and random seed 42. Experimental results show that the original YOLOv11n baseline achieves the highest mAP@0.5 and mAP@0.5:0.95, outperforming all attention variants. The three best-performing attentions, namely ELA, TripletAttention, and SimAM, all achieve a mAP@0.5 of 0.735, only 0.003 lower than the baseline. In contrast, CBAM and SE, which have more parameters, drop by 0.066 and 0.076, respectively, and training curve analysis confirms that the degradation stems from overfitting. Per-class analysis reveals that crazing and rolled-in scale are the most challenging defect categories, with no attention mechanism improving detection accuracy on these classes over the baseline. The parameter-free SimAM shows the smallest drop, supporting the view that lightweight designs are safer on small datasets. Supplementary experiments at 640×640 resolution further validate the superiority of the baseline (baseline 0.772 vs. SimAM 0.761, ELA 0.741, SE 0.684). All models meet the real-time detection requirements. This baseline provides the real-world results of model selection for small industrial defect samples: a good baseline is often enough, attention should be emphasized, and parameter free or low-parameter designs should be chosen.

Keywords: Strip Surface Defect Detection; YOLOv11n; Attention Mechanism; NEU-DET; Benchmark; Real-time Detection; Overfitting.

1. Introduction

Strip steel is widely used in industries such as automotive, construction, and home appliances, and its surface quality directly affects the performance and service life of final products. During the production of strip steel, due to factors such as mechanical wear of rolling equipment, mixing of raw material impurities, and fluctuations in process parameters, various defects inevitably appear on the strip surface, including crazing, inclusion, patches, pitted surface, rolled-in scale, and scratches [1-3]. If these defects are not detected and addressed in a timely manner, they will lead to increased product scrap rates, higher production costs, and even safety accidents in severe cases. Therefore, strip surface defect detection is a key link in quality control for the steel industry.

Traditional strip surface defect detection mainly relies on manual visual inspection, which has many limitations: low detection efficiency, making it difficult to adapt to the rhythm of modern high-speed rolling production lines; strong subjectivity, as inspection results are affected by factors such as the inspector's experience, attention, and fatigue, making consistency and reliability difficult to guarantee; and insufficient sensitivity to small defects, leading to missed detections and false alarms [2]. New solutions for strip surface defect detection are provided by computer vision, especially deep learning technology. Deep learning methods can automatically learn defect features and realize automated detection of defect locations, categories, and severities, thereby greatly enhancing inspection efficiency and accuracy [3,4].

Among object detectors, the YOLO series is popular because it is fast and accurate. YOLOv1 to YOLOv11 are all more recent, and each iteration has brought significant updates to the design of the model, features, etc. The new YOLOv11n is a small backbone network that has been improved with new modules, such as C2f, C2PSA and an anchor-free detection head; it is fast in operation and only has 2.58M parameters and 6.3 GFLOPs [5]. Its design makes it an ideal baseline model for attention integration.

Attention mechanisms have been widely used to enhance the feature representation capability of detection networks. Modules such as Squeeze-and-Excitation (SE) [6], Convolutional Block Attention Module (CBAM) [7], and parameter-free SimAM [8] have shown improvements on large-scale datasets (e.g., COCO). However, their effectiveness on small industrial datasets (where data is scarce and defect sizes vary) has not been fully investigated. Most existing work focuses on proposing new attention designs or applying them in a single manner, lacking systematic and fair comparisons under identical conditions [9,10].

To fill this gap, we conduct a comprehensive benchmark of 15 attention mechanisms integrated into YOLOv11n on the NEU-DET strip defect dataset. The main contributions of this paper are as follows: (1) We systematically evaluate 15 attention modules, including channel, spatial, hybrid, and parameter-free types, under strictly identical training

protocols for the first time. (2) The original YOLOv11n baseline gave us the best mAP@0.5 score of 0.738. It did better than every attention-based model we tested. Some complex attentions like CBAM and SE fell off a lot because they overfit. We also broke down the AP numbers for each defect type and reported efficiency metrics like FPS, parameter count, and GFLOPs, so others can use this as a reference for real deployment. On top of that, we ran extra tests at 640×640 resolution, and the baseline still came out on top, which shows our result is robust. Putting all this together, adding attention modules to a small defect dataset might not help at all, or could even hurt performance. Hopefully this stops future researchers from throwing attention mechanisms at their models without a good reason.

2. Literature Review

2.1. Strip Surface Defect Detection

People started working on strip surface defect detection a long time ago using handcrafted features. Back then, they mostly used traditional image processing like edge detection, thresholding, and texture analysis. But these methods don't generalize well when the background is complex or the defects come in many shapes, so they couldn't really meet what industry needed [1]. Then deep learning came along, and CNN-based methods took over. At first, deep learning was mostly used for classification—just telling whether a defect is there and what type it is, but not where it is or how big [2].

After that, people brought object detection into strip defect detection. Two-stage detectors like Faster R-CNN use an RPN to propose regions and then do classification and regression with RoI Pooling. They improved accuracy a lot, but they run slow, so real-time detection is hard [3]. On the other hand, one-stage detectors like SSD and the YOLO family treat detection as a regression problem. That makes them fast and end-to-end, and they strike a decent balance between speed and accuracy [4]. The YOLO series, from YOLOv3 up to YOLOv11, has worked really well for strip defect detection.

There have also been many YOLO-based improvements for the NEU-DET dataset. For instance, Mo Zhenkun [1] looked into lightweight models for detecting strip surface defects, making them more efficient by compressing the model and tweaking the structure. Sun et al. [2] came up with YOLO-AMM, a lightweight algorithm that uses multi-module attention fusion to get better accuracy. Yang Quanchao [3] did a deep dive into lightweight deep learning models for this task. Dai et al. [4] proposed HSED-YOLO, which cuts down model complexity a lot while keeping accuracy high. Xu et al. [11] designed a steel defect detection algorithm based on an improved YOLOv11n, boosting small-target detection with multi-scale feature fusion and attention. Lei et al. [12] introduced DFD-YOLOv11n, which improves detection accuracy by optimizing both feature extraction and the detection head. He et al. [13] presented an efficient steel surface defect detection method built on a lightweight YOLO framework, balancing speed and accuracy with task-specific knowledge-guided optimization. But here's the thing—most of these studies focus on coming up with new methods, not on systematic comparisons. They don't offer a fair evaluation of different attention mechanisms under the same framework.

2.2. Evolution of the YOLO Series

YOLOv11n is the newest lightweight model in the YOLO family. It keeps the efficient backbone design and adds a module called C2PSA (Cross Stage Partial Spatial Attention). With only 2.58 million parameters and 6.3 GFLOPs, it still pulls off efficient detection. Fig. 1 shows how the whole network is laid out.

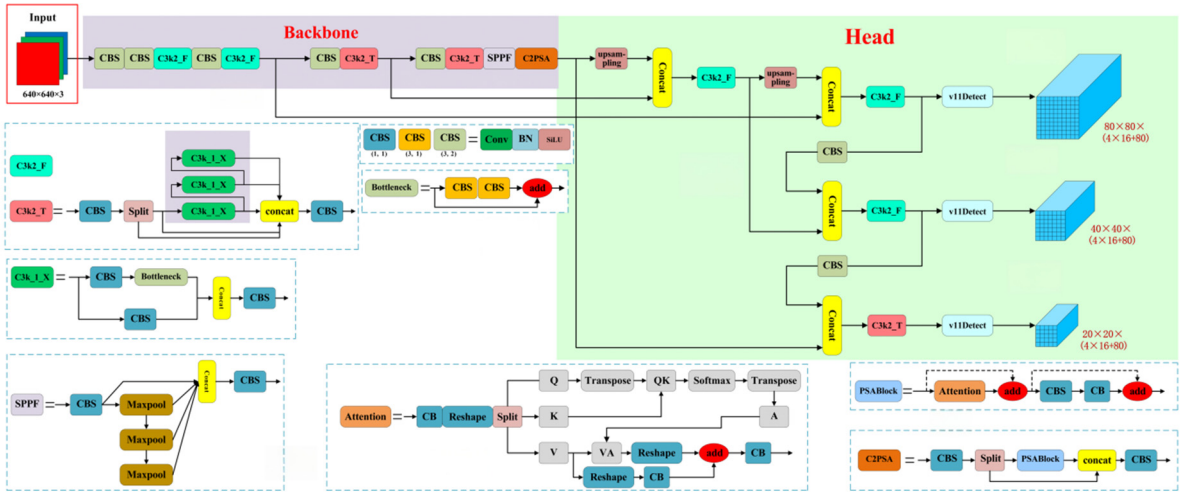

Fig 1. A schematic picture of how the entire YOLOv11n network is put together

Fig. 1 shows what's going on inside the network. The backbone has a C2PSA module that uses spatial attention to pull out better features. The neck mixes features from different levels, and the detection head gives the final outputs. So the whole design is light but still uses spatial attention to get good features. That's exactly why this model works well as a baseline for attention experiments—it already has a spatial attention piece built in, so we can test if adding extra attention modules helps or not [5]. Also, Sun et al. [5] used YOLOv11n to build a lightweight chip defect detection method, and Wang et al. [14] came up with a lightweight bearing defect detection algorithm based on an improved YOLOv11. Both studies confirmed that YOLOv11n has potential for industrial defect detection.

2.3. Attention Mechanisms in Object Detection

The core idea behind attention mechanisms is to replicate the selective focus of the human visual system. In practice, this allows a network to assign more computation to important feature regions while reducing the influence of

irrelevant or redundant information. Attention mechanisms can be grouped by the dimension they operate on and the way they work.

Channel attention. Typical examples are SE [6] and ECA [9]. SE uses global average pooling and fully connected layers to learn relationships between channels and then reweights each channel. ECA uses one-dimensional convolution for cross-channel interaction, achieving similar performance with fewer parameters.

Spatial attention. This type focuses on where features are located within a feature map. The Spatial Attention Module (SAM) is a representative method. It generates attention weights along spatial directions, helping the network concentrate on key parts of a target.

Hybrid attention. These methods combine channel and spatial attention. Representative modules include CBAM [7], GAM, BAM, and ACmix. CBAM applies channel attention first and then spatial attention to refine the features. GAM enhances feature representation by retaining and interacting with global information.

Parameter-free attention. SimAM [8] is the main example. It is derived from an energy function framework in neuroscience. It computes attention weights by measuring the difference between a feature and its context, without introducing any trainable parameters. This design makes it less prone to overfitting on small datasets.

Coordinate attention. CA [10] is an example. It applies global average pooling along the two spatial directions separately and then uses one-dimensional convolutions to encode coordinate information, capturing both channel relationships and long-range dependencies.

In addition, many newer attention mechanisms have been proposed, including EMA, SK, TripletAttention, A2, ELA, SLAM, and SCSA. Most of these have been evaluated on large datasets such as COCO or ImageNet, where the large amount of data helps reduce overfitting. However, their performance on small datasets like NEU-DET (only 1,800 images) remains unclear. Our work addresses this gap by providing a fair, side-by-side comparison.

3. Experimental Setup

3.1. Dataset

Researchers commonly use the NEU-DET dataset as a benchmark for strip surface defect detection. It was created by Northeastern University and contains 1,800 grayscale images at 200×200 pixels. The dataset covers six defect types—crazing, inclusion, patches, pitted surface, rolled-in scale, and scratches—with 300 images per class. Several characteristics define this dataset: image resolution is low, defect sizes vary a lot, and there are notable intra-class differences as well as similarities across different classes. These properties make it well suited for testing detection algorithms [1-3].

We split the dataset randomly, taking 70% (1,260 images) for training and the remaining 30% (540 images) for validation. Because the dataset is small and this study is exploratory, we did not create a separate test set. Every model uses exactly the same split, so the relative comparisons are fair. We set the random seed to 42 to make the results reproducible.

3.2. Evaluation Metrics

We used several metrics to get a full picture of each model's performance. (1) Precision: out of all the samples a model labeled as positive, how many were actually positive. (2) Recall: out of all the truly positive samples, how many did the model correctly catch. (3) Mean average precision: mAP@0.5 is the average precision when the IoU threshold is 0.5, and mAP@0.5:0.95 is the average over IoU thresholds from 0.5 to 0.95 in steps of 0.05. (4) Model size: we looked at the number of parameters and GFLOPs. (5) Inference speed in FPS: we measured this on an NVIDIA RTX 3060 laptop GPU with a batch size of 1.

3.3. Implementation Details

We trained every model with the same hyperparameters so the comparison would be fair. Here's what we used: hardware was an NVIDIA GeForce RTX 3060 GPU with 6GB memory. Software included Python 3.10, PyTorch 2.0.1, and Ultralytics 8.3.185. We ran training for 50 epochs, with early stopping patience set to 15. Batch size was 16. The main experiments used 200×200 input resolution, and we also did a robustness check at 640×640. The optimizer was AdamW, with learning rate 0.001 and weight decay 0.0005. For data augmentation, we applied Mosaic (probability 1.0) and left MixUp and Copy-Paste at their defaults (0.0). The random seed was fixed to 42.ll models are trained using identical hyperparameters to ensure fairness and comparability. The specific configuration is as follows: hardware environment: NVIDIA GeForce RTX 3060 GPU with 6GB memory; software environment: Python 3.10, PyTorch 2.0.1, Ultralytics 8.3.185; number of training epochs: 50, early stopping patience: 15; batch size: 16; input resolution for the main experiment: 200×200, and 640×640 for robustness check; optimizer: AdamW, learning rate: 0.001, weight decay: 0.0005; data augmentation: Mosaic, while MixUp and Copy-Paste are kept at their default values; random seed is fixed to 42.

3.4. Attention Integration Mechanism

Each attention module is inserted into the backbone or neck of YOLOv11n according to its original design. Specifically, channel attention modules such as SE and ECA are inserted inside the C2f module to reweight the channel dimension after feature extraction. For spatial attention, take SimAM as an example—it goes between feature maps and adjusts the importance of different spatial locations on the fly. As for hybrid attention, modules like CBAM and GAM deal with both channels and spatial positions at the same time. The configurations are defined in YAML files, for example yolo11-SimAM.yaml, and loaded via the Ultralytics framework. All models are initialized with the yolo11n.pt pretrained weights to ensure consistency at the start of training.

4. Experimental Results and Discussion

4.1. Overall Performance Analysis

Table 1 reports the best validation metrics for all 16 models. The original YOLOv11n baseline achieves the highest mAP@0.5 and mAP@0.5:0.95. Three attention mechanisms ELA, TripletAttention, SimAM reach 0.735, only 0.003 lower than the baseline. In contrast, CBAM and SE drop by 0.066 and 0.076, respectively, representing significant degradation. In terms of precision and recall, the baseline exhibits a good balance: precision of 0.723 and recall of 0.660. Notably, GAM achieves the highest precision but with a lower mAP,

indicating that this model tends to make more conservative predictions (high precision means a high proportion of correctly predicted positive instances among those predicted as positive, but may miss many positives). TripletAttention achieves the highest recall, indicating more comprehensive detection coverage, albeit with slightly lower precision than the baseline.

Table 1. Performance comparison of YOLOv11n and 15 attention mechanisms on the NEU-DET validation set

Model	mAP@0.5	mAP@0.5:0.95	Precision	Recall	FPS
Baseline (YOLOv11n)	0.738	0.418	0.723	0.660	125.3
ELA	0.735	0.400	0.680	0.677	—
TripletAttention	0.735	0.393	0.682	0.694	—
SimAM	0.735	0.396	0.662	0.688	110.6
SLAM	0.732	0.403	0.679	0.678	—
GAM	0.727	0.393	0.749	0.654	—
CA	0.727	0.399	0.690	0.675	—
ACmix	0.726	0.407	0.713	0.670	—
EMA	0.726	0.397	0.702	0.671	121.8
ECA	0.715	0.381	0.711	0.636	—
SCSA	0.713	0.383	0.660	0.684	—
SK	0.712	0.371	0.708	0.638	—
A2	0.708	0.369	0.659	0.677	—
BAM	0.704	0.369	0.711	0.618	—
CBAM	0.672	0.349	0.679	0.617	117.2
SE	0.662	0.335	0.654	0.616	119.8

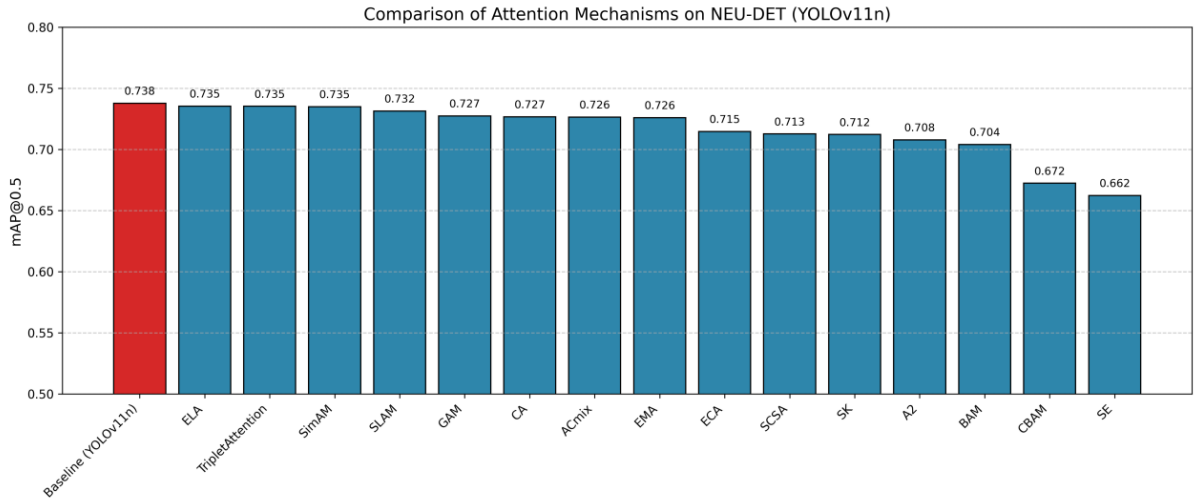

Fig 2. Comparison of mAP@0.5 between YOLOv11n and 15 attention mechanisms on the NEU-DET validation set

As can be observed from Table 1 and Fig. 2, the mAP@0.5 of all attention variants does not exceed that of the baseline,

indicating that simply adding attention modules on a small industrial dataset such as NEU-DET does not bring performance improvement and may even lead to overfitting due to extra parameters, e.g., CBAM and SE. Meanwhile, attention performance is not proportional to complexity: the parameter-free SimAM and lightweight ELA and TripletAttention are closest to the baseline, while CBAM and SE, which have more parameters, drop significantly, suggesting that lightweight or parameter-free attentions should be preferred for small datasets. Furthermore, there is a trade-off between precision and recall: the baseline achieves a good balance; GAM has the highest precision but lower mAP, indicating more conservative predictions; TripletAttention achieves the highest recall but slightly lower precision, offering more comprehensive detection at the cost of more false positives. So different attention mechanisms end up affecting the network's decision boundary in different ways.

4.2. Per-Class Analysis

Take a look at Fig. 3. It's a heatmap that shows the average precision (AP) for each model on every defect class. From this heatmap, we can see a few things.

Some defects are much easier to detect than others. Patches turned out to be the easiest, with the baseline model hitting an AP of 0.612. That's probably because patches are usually large and have strong texture patterns, so the network picks them up without much trouble. Scratches and inclusion are somewhere in the middle—baseline APs of 0.526 and 0.450 respectively. Pitted surface also falls into that medium range, with an AP of 0.478.

On the other hand, crazing and rolled-in scale are the hardest ones. Their baseline APs are only 0.171 and 0.271. Why is crazing so low? Its features are thin and spread out all over, and they look a lot like normal surface texture, so the network gets confused. Rolled-in scale has low contrast against the normal surface and comes in many different shapes, which also makes detection difficult. The low AP for rolled-in scale may be due to its low contrast with normal surfaces and its diverse morphological variations. These findings are consistent with previous studies [1-3], indicating that the class imbalance issue in the NEU-DET dataset requires special attention.

More importantly, no attention mechanism improves the AP for crazing beyond the baseline; ELA is the closest, yet still 0.004 lower. SE performs the worst across all classes, especially on crazing and rolled-in scale, confirming that complex attentions not only fail to help but actually harm the detection of difficult defects. This phenomenon may be explained by the fact that the extra parameters introduced by complex attention modules learn low-level noise patterns during training, rather than features that truly help distinguish hard samples.

In addition to quantitative analysis, we visually compare the detection results of representative models. As shown in Fig. 4, detection examples of the baseline (YOLOv11n), the best attention variant (SimAM), and the worst attention variant (SE) on the six defect classes are presented. It can be seen that the baseline model achieves the most accurate localization for all defect types; SimAM performs similarly to the baseline; while SE exhibits obvious missed detections and false positives on challenging defects such as crazing and rolled-in scale, consistent with the quantitative result that SE has the lowest AP in Fig. 3.

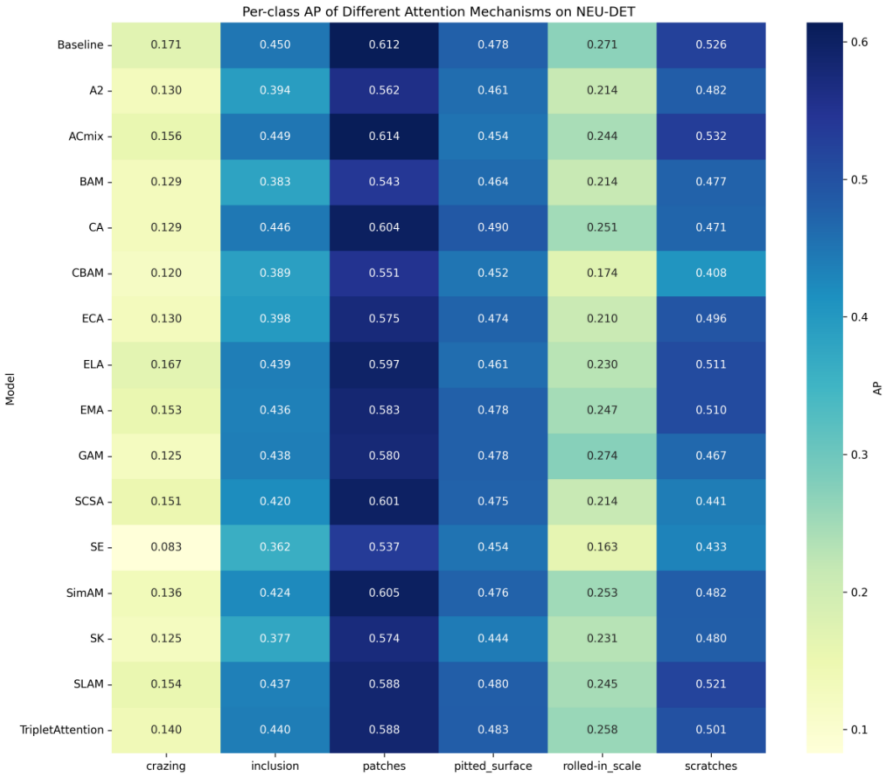

Fig 3. Heatmap of AP per defect class for all models

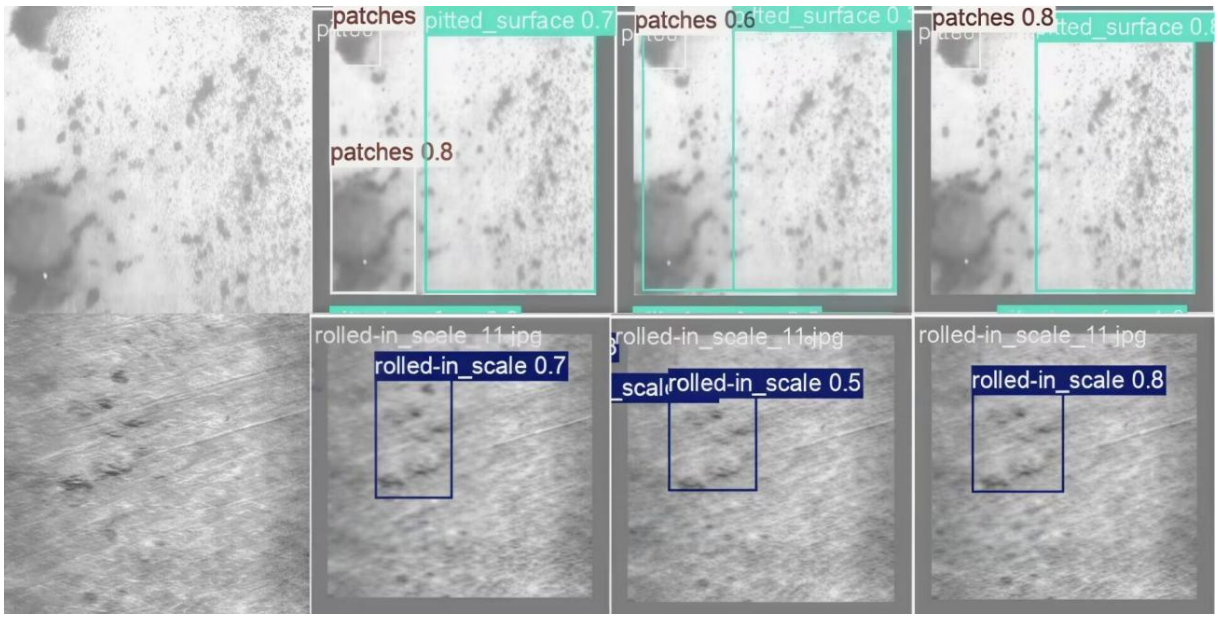

Fig 4. Comparison of defect detection results of baseline, SimAM, and SE models on the NEU-DET dataset

4.3. Training Dynamics and Overfitting Analysis

To gain a deeper understanding of the impact of attention mechanisms on the training process, we compare the training dynamics of four representative models: baseline, SimAM, CBAM, and SE. As shown in Fig. 5, the left subplot presents the validation mAP@0.5 curve, and the right subplot presents the validation classification loss curve.

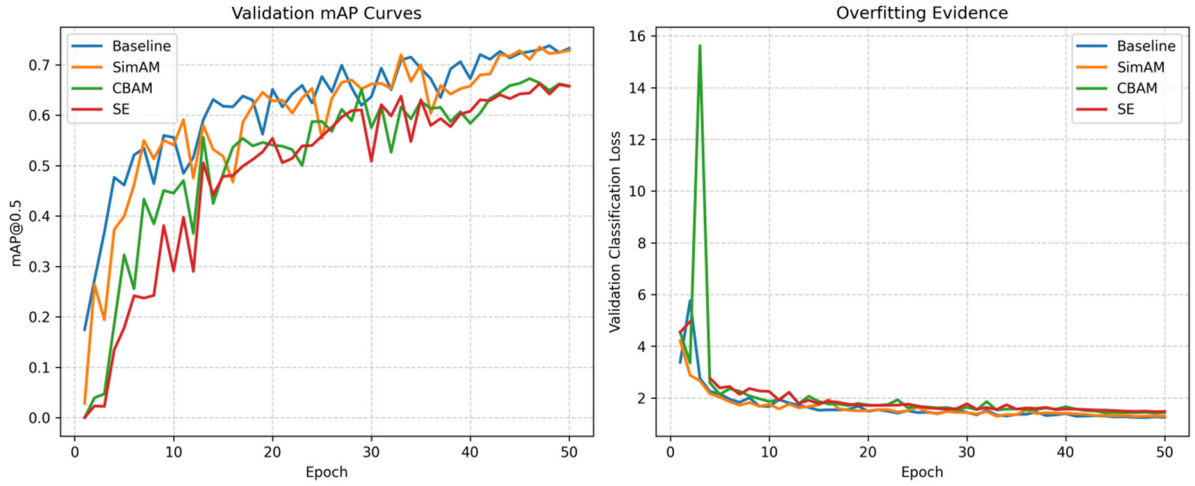

Fig 5. Training dynamics and overfitting analysis of four representative models on the NEU-DET validation set

As can be seen from the left subplot of Fig. 5, the baseline model converges rapidly and remains stable, reaching a high level after approximately 30 epochs and maintaining it until the end of training. The convergence behavior of SimAM is highly similar to that of the baseline, which is consistent with its parameter-free design: it introduces no additional trainable parameters and therefore does not alter the optimization trajectory of the model.

On the other hand, the mAP curves for CBAM and SE jump around quite a bit. Looking at the validation loss curves on the right side of Fig. 5, after about 40 epochs the loss for both CBAM and SE starts climbing steadily. That is a clear sign of overfitting. The reason can be traced to the parameter count. CBAM adds roughly 10,000 extra parameters, and SE adds about 16,000. With only 1,260 training images, those extra parameters make the models fit too closely to the training data, so they do not generalize well to the validation set. This matches the classic deep learning observation that overfitting becomes likely when the number of parameters exceeds the amount of data.

Even with early stopping turned on, CBAM and SE still get noticeably worse later in training. Therefore, early stopping alone is not enough to fix the overfitting problem caused by attention modules on small datasets. For small industrial datasets, a better approach is to choose attention mechanisms with few or no parameters, or to apply stronger regularization.

4.4. Efficiency Analysis

All models use the same backbone, which is YOLOv11n. How much extra computation an attention module brings depends on how it is designed. SimAM has no parameters at all—it generates attention weights just by computing the energy function of the feature maps. CBAM adds roughly 0.5% more parameters, so the extra cost is small. ELA, on the other hand, increases the parameter count by about 90%, pushing it up to 4.90M. That is a heavy additional burden.

ll models share the same YOLOv11n backbone. The computational overhead introduced by attention modules varies depending on the module design. SimAM, as a parameter-free attention mechanism, adds no additional parameters and generates attention weights solely by computing the energy function of the feature maps. CBAM adds approximately 0.5% more parameters, incurring a small overhead. Due to its design characteristics, ELA increases the parameter count by approximately 90%, reaching 4.90M, which represents a significant extra burden.

Table 2. Inference speeds of representative models (RTX 3060, batch=1)

Model	FPS	Parameters (M)	GFLOPs
Baseline (YOLOv11n)	125.3	2.58	6.3
SimAM	110.6	2.58	6.3
CBAM	117.2	2.59	6.4
EMA	121.8	2.60	6.4
SE	119.8	2.60	6.3

Table 2 shows the numbers. All models exceed 100 FPS, meaning they satisfy real-time detection. The baseline reaches the highest speed at 125.3 FPS, while SimAM is the lowest at 110.6 FPS. SimAM introduces no extra parameters, yet its energy function computation still creates some inference overhead. Altogether, attention modules have only a small effect on inference speed and do not prevent real-time deployment.

4.5. Robustness to Input Resolution

We trained four models (baseline, SimAM, ELA, and SE) at 640×640 resolution to check whether the baseline's advantage continues to hold with larger input sizes.Table 3 shows the results: the baseline again achieves the highest mAP, while SimAM, ELA, and SE all lag behind. Notably, ELA has 4.90M parameters and 8.5 GFLOPs, nearly doubling the model size, yet its performance remains inferior to the baseline. This further confirms that the observed phenomenon is independent of the resolution setting.

Table 3. Experimental results at 640×640 input resolution

Model	mAP@0.5	Parameters (M)	GFLOPs
Baseline	0.772	2.58	6.3
SimAM	0.761	2.58	6.3
ELA	0.741	4.90	8.5
SE	0.684	2.58	6.3

Comparing the results at 200×200 and 640×640 resolutions, it can be observed that all models achieve performance improvements at the higher resolution, because the higher resolution provides richer spatial detail information. However, the advantage of the baseline remains stable under both resolutions, indicating that our findings are robust. In particular, the performance drop of SE is significant under both resolutions, further confirming the adverse properties of complex attentions on small datasets.

4.6. Discussion

4.6.1. Why is the Baseline Better Than Attention

On a small dataset, the model capacity of YOLOv11n is already sufficient. YOLOv11n itself already contains the C2PSA spatial attention module, providing a certain spatial modeling capability. Adding extra attention modules introduces additional parameters or complex operations, leading to overfitting when data is insufficient. The parameter-free SimAM shows the smallest drop, supporting the view that lightweight designs are safer. Moreover, the insertion location of attention modules may conflict with the optimization of the original architecture. The architecture of YOLOv11n has been carefully designed and optimized, and inserting additional modules may disrupt the original feature flow patterns.

4.6.2. Comparison with Previous Works

Xu et al.[15] proposed a lightweight steel surface defect detection method by improving YOLOv8, and Gang et al.[16] improved the lightweight algorithm of YOLOv8n. These works achieved performance gains through multi-dimensional optimizations, but none of them separately evaluated the contribution of attention mechanisms. Our results complement these works by showing that a single attention mechanism cannot outperform the baseline without other architectural changes.

4.6.3. Practical Recommendations

Based on the experimental results of this study, we provide the following recommendations for model selection on small industrial defect datasets. For small industrial defect datasets, it is advisable to use the original YOLOv11n without adding extra attention. If attention must be used, parameter-free types such as SimAM should be prioritized, followed by low-parameter attentions such as ELA and TripletAttention. Complex attentions like CBAM and SE should be avoided, as their risk of overfitting on small datasets far outweighs any potential performance gain. If performance improvement is indeed required, it is recommended to adopt a multi-dimensional approach, including data augmentation, architectural optimization, loss function improvement, etc., rather than relying solely on attention mechanisms.

4.6.4. Limitations

This study has the following limitations. First, we did not use a separate test set; all reported metrics are based on the validation split. This may lead to slightly optimistic results, but the relative comparison remains fair as all models share the same split. Second, we conducted experiments on only one dataset, NEU-DET. The generalizability of our conclusions needs to be verified on more datasets. Third, the insertion location of attention modules may affect the results. We followed the original recommendation for placing each attention module, but we did not systematically examine how different insertion positions affect performance. Future research may validate these findings on larger and more

diverse datasets, and may also develop adaptive attention mechanisms specifically for small-scale domains.

5. Conclusion

TWe evaluated 15 attention mechanisms integrated into YOLOv11n for strip surface defect detection on the NEU-DET dataset. Under identical training conditions, the original YOLOv11n baseline achieved the highest mAP@0.5, outperforming all attention variants. The three best performers (ELA, TripletAttention, SimAM) reached only 0.735, whereas CBAM and SE dropped significantly due to overfitting. Per-class analysis showed that crazing and rolled-in scale are the most difficult defects, and none of the attention mechanisms improved detection on these classes.

Supplementary experiments at 640×640 resolution confirmed the baseline's superiority, meaning the observed phenomenon does not depend on resolution. All models ran at real-time FPS, and attention modules had only a small effect on inference speed. The key message for small defect datasets is that a strong baseline is usually sufficient, and adding attention should be done with caution.

Acknowledgments

The authors are grateful to the providers of the NEU-DET dataset. Thanks also go to the supervising advisor for guiding and assisting the research, and to fellow students for their support and discussions during the writing of this paper.

References

[1] Mo, Z. K. (2025). *Research on Lightweight Model for Strip Surface Defect Detection* (Master's dissertation). Hangzhou Dianzi University.

[2] Sun, Y. L., & Chen, G. F. (2026). Lightweight Strip Surface Defect Detection Algorithm Based on YOLO-AMM. *Electronic Measurement Technology*. Advance online publication.

[3] Yang, Q. C. (2025). *Research on Lightweight Deep Learning Model for Strip Surface Defect Detection* (Master's dissertation). Shijiazhuang Tiedao University.

[4] Dai, L. H., Li, Y. S., Shi, R., et al. (2025). HSED-YOLO: A Lightweight Strip Surface Defect Detection Model. *Journal of Guangxi Normal University (Natural Science Edition)*, 43(2), 95–106.

[5] Sun, P. Y., Huang, J., Gu, J. A., et al. (2025). Lightweight Chip Surface Defect Detection Method Based on YOLO11n. *Semiconductor Technology*, 50(7), 707–713.

[6] Hu, J., Shen, L., & Sun, G. (2018). Squeeze-and-Excitation Networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 7132–7141). IEEE. https://doi.org/10.1109/CVPR.2018.00745.

[7] Woo, S., Park, J., Lee, J. Y., & Kweon, I. S. (2018). CBAM: Convolutional Block Attention Module. In *Proceedings of the European Conference on Computer Vision (ECCV)* (pp. 3–19). Springer. https://doi.org/10.1007/978-3-030-01234-2_1.

[8] Yang, L., Zhang, R. Y., Li, L., & Xie, X. (2021). SimAM: A Simple, Parameter-Free Attention Module for Convolutional Neural Networks. In *Proceedings of the 38th International Conference on Machine Learning (ICML)* (pp. 11863–11874). PMLR. https://doi.org/10.48550/arXiv.2105.15133.

[9] Wang, Q., Wu, B., Zhu, P., Li, P., Zuo, W., & Hu, Q. (2020). ECA-Net: Efficient Channel Attention for Deep Convolutional Neural Networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 11534–11542). IEEE. https://doi.org/10.1109/ CVPR42600. 2020.01155.

[10] Hou, Q., Zhou, D., & Feng, J. (2021). Coordinate Attention for Efficient Mobile Network Design. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 13713–13722). IEEE. https://doi. org/ 10.1109/ CVPR46437.2021.01350.

[11] Xu, J., Jia, Y. A., & Wang, Z. F. (2026). Steel Surface Defect Detection Algorithm Based on Improved YOLOv11n. *Laser & Optoelectronics Progress*, 63(2), 205–216.

[12] Lei, F. Q., Ma, L. W., Guan, P., et al. (2026). Research on Steel Equipment Surface Defect Detection Algorithm Based on DFD-YOLOv11n. *Computer Engineering and Applications*. Advance online publication.

[13] Xu, H., Zhang, Z., Ye, H., et al. (2025). Efficient Steel Surface Defect Detection via a Lightweight YOLO Framework with Task-Specific Knowledge-Guided Optimization. *Electronics*, 14, 2029. https://doi.org/10.3390/electronics14102029.

[14] Wang, Y. P., Wei, Z., & Zhang, K. (2026). Lightweight Bearing Surface Defect Detection Algorithm Based on Improved YOLOv11. *Journal of Computer Applications*. Advance online publication.

[15] Xu, J. M., Cao, S., & Guan, H. Y. (2025). Lightweight Steel Surface Defect Detection Method Based on Improved YOLOv8. *Electronic Measurement Technology*, 48(24), 138–147.

[16] Gang, S., Liu, P. S., & Guo, X. W. (2025). Improved Lightweight Steel Surface Defect Detection Algorithm Based on YOLOv8n. *Electronic Measurement Technology*, 48(3), 74–82.