Journal of Computing and Electronic Information Management
ISSN: 2413-1660 | Vol. 20, No. 3, 2026

A lightweight bauxite detection and identification algorithm

Hongyu Zhang

School of Computer Science and Technology, Taiyuan Normal University, Jinzhong 030619, China

Abstract: To address the issues of low efficiency and poor accuracy of manual methods in bauxite sorting, this paper proposes a lightweight ore identification and detection algorithm, CL-YOLOv11, based on an improved YOLOv11. This algorithm incorporates a C3k2-GD module integrating GhostModule and dynamic convolution in the backbone network to enhance adaptive feature extraction while reducing redundant parameters. Furthermore, a lightweight shared detail enhancement detection head (LSDECD) is constructed in the detection head, improving the accuracy of capturing small targets and texture edges by introducing detail enhancement convolution and learnable dynamic factors. Experimental results show that compared to the original model, CL-YOLOv11 reduces the number of parameters by approximately 33%, decreases floating-point computation by 23%, and significantly improves inference speed. It achieves a balance between high accuracy and high efficiency while maintaining a high detection accuracy of 93.60%, providing an effective technical solution for automated intelligent bauxite sorting.

Keywords: Bauxite; Target detection; YOLOv11.

1. Introduction

Bauxite, also known as bauxite, is a general term for ores composed mainly of gibbsite, boehmite and diaspore. It is a non-renewable resource. Bauxite has extremely high economic value, and its uses are mainly divided into two major fields: metals and non-metals. In the metals field, more than 90% of the worlds bauxite production is used in the aluminum smelting industry. Usually, mature processes such as the Bayer process or sintering process are used to convert the selected bauxite into the intermediate product alumina. Subsequently, the alumina is electrolyzed to finally obtain metallic aluminum, which is then cast into various aluminum ingots and aluminum alloy billets. It is not only widely used in traditional fields such as building doors and windows and packaging materials, but also a core material for lightweight structural parts of aerospace vehicles, lightweight body and battery shells of new energy vehicles, high-voltage transmission lines in the power industry, and electronic and electrical products.

Bauxite also has wide applications in the non-metallic materials sector. After calcination or chemical treatment, bauxite can be used to produce refractory materials such as high-alumina bricks and castables. It is also an important raw material for the preparation of aluminum chemical products such as alumina, aluminum hydroxide, and aluminum chloride. These compounds play a key role in flame retardants, water purification agents, papermaking additives, and catalysts. The corundum-like materials obtained from calcined bauxite can be used as high-hardness abrasives to manufacture grinding wheels, blasting media, and polishing materials. Some bauxite can be used in the production of high-alumina cement and as environmentally friendly adsorbent materials, such as activated alumina, in water treatment and gas purification. Bauxite possesses irreplaceable economic value.

Bauxite resources involve a complex development process, primarily including ore crushing, washing and beneficiation, smelting, and processing. The beneficiation process is a crucial step in the entire resource development process; the accuracy and efficiency of beneficiation directly affect the grade of bauxite and whether the comprehensive utilization rate of resources can be optimized. Currently, bauxite beneficiation mainly employs traditional manual methods. This method is not only slow and inaccurate, but also easily influenced by subjective human factors. In recent years, deep learning technology has developed rapidly, enabling the use of deep learning object detection algorithms for bauxite beneficiation. Compared to manual methods, deep learning can significantly improve beneficiation efficiency and accuracy, effectively reduce the interference of subjective factors, and establish objective and unified beneficiation standards.

2. Related Work

Zhen J et al[1] proposed an OreYOLO ore sorting network based on a fusion attention mechanism for sorting gold and pyrite. This method introduces an efficient multi-scale attention mechanism (EMA) on top of YOLOv5 to enhance feature extraction capabilities; simultaneously, it employs an asymptotic feature pyramid network (AFPN) for multi-scale feature fusion to avoid information loss and optimizes the network into a lightweight structure. Through comparison with mainstream detection algorithms, a balance between high accuracy and efficient deployment is achieved. Attallah et al[2]. proposed an optimized hybrid convolutional neural network (CNN) model for automatic mineral classification in hyperspectral images. They used diagnostic absorption band (DAB) selection technology for band simplification to extract key mineral absorption features; designed and optimized a 3D-2D hybrid CNN architecture to efficiently capture spatial and spectral features, successfully identifying eight minerals including alunite and calcite. Choros K Aet al[3]. combined hyperspectral imaging with neural network classification algorithms for in-situ ore grade determination. Mohammed Q. Alkhatib[4] proposed SS-MixNet, a lightweight deep learning model for hyperspectral image classification. It employs 3D convolution to extract local spectral spatial features, a parallel

MLP mixer to capture long-range dependencies, and introduces a deep convolutional attention mechanism to enhance discriminative power. Using only 1% of the labeled data, the model achieved overall accuracies of 95.68% and 93.86% on the QUH-Tangdaowan and QUH-Qingyu datasets, respectively, validating the effectiveness and robustness of the algorithm.

While existing research has made some progress in the field of ore identification, shortcomings remain. For example, OreYOLO improves detection performance by introducing an attention mechanism, but its model structure is complex and computationally expensive. Hyperspectral methods based on 3D-2D hybrid CNN models, while achieving high classification accuracy, rely on expensive equipment and suffer from poor real-time performance. In summary, existing methods still have room for improvement in terms of lightweight design, real-time performance, and detection capabilities. Therefore, this paper proposes a lightweight target detection algorithm based on improved YOLOv11 to achieve a balance between high accuracy and high efficiency.

3. Model Improvement

YOLOv11, as a lightweight, single-stage object detection model, inherits the high-efficiency detection characteristics of the YOLO series, possessing advantages in both model size and inference efficiency. However, in the complex application scenario of bauxite sorting, where ore surface textures vary greatly, shapes are irregular, and target scales span a wide range, YOLOv11s feature representation capabilities and multi-scale information capture capabilities still have shortcomings. For example, the model has a limited range of perception for small-scale ore targets and local detail features, leading to false positives and false negatives in complex backgrounds. Furthermore, its feature fusion structure has room for optimization in terms of multi-scale feature utilization and computational resource allocation, and it cannot fully meet the actual requirements of high-precision sorting tasks.

To address the aforementioned issues, this paper, considering the characteristics of bauxite sorting tasks, makes targeted improvements and optimizations to the YOLOv11 network model at both the backbone network and the detection head levels, proposing a target detection algorithm based on CL-YOLOv11. In the backbone network, based on the ParameterNet[5] design concept, GhostModule and Dynamic Convolution are combined to optimize the original C3k2 module, resulting in an improved C3k2-GD module. This module enhances the networks feature extraction capabilities while effectively reducing redundant parameters and computational load, thereby lowering the overall complexity while maintaining the models expressive power. Regarding the detection head design, this paper proposes a lightweight shared detail-enhanced convolutional detection head, LSDECD (Lightweight Shared Detail-Enhanced Convolutional Detection). Through shared parameters and detail enhancement mechanisms, it improves the detection heads ability to capture local texture and edge information, further enhancing detection accuracy while reducing computational overhead.

3.1. C3k2-GD Model

To address the issues of large number of parameters and high computational cost in traditional convolution operations of YOLOv11, this paper proposes GhostModule to optimize and improve the C3k2 feature extraction module. This module utilizes a linear transformation method to efficiently generate additional feature maps based on existing feature maps. Traditional methods require additional convolutional layers to generate additional feature maps. In contrast, the GhostModule modification scheme can reduce computational resource consumption[6].

GhostModule first uses a two-dimensional convolutional layer combined with batch normalization and ReLU activation function to extract basic features and adjust the number of channels in the input feature map. Then, it generates a supplementary feature map through linear transformation. Finally, it fuses the main feature map generated by traditional convolution operation with the auxiliary feature map obtained by linear transformation to obtain the final output feature. The internal structure of GhostModule is shown in Figure 1.

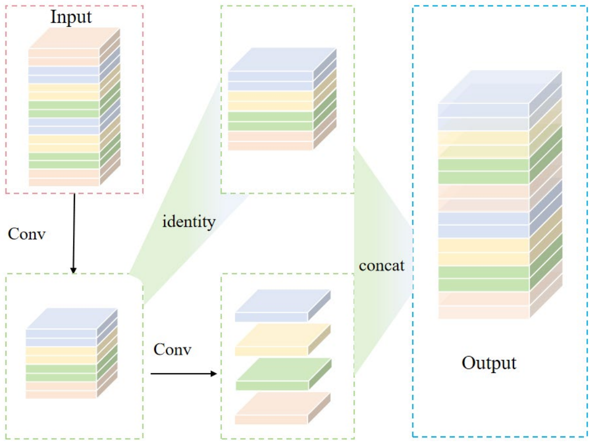

Fig 1 Internal structure diagram of GhostModule

GhostModule can effectively reduce the number of model parameters and floating-point operations. Using this idea in large-scale visual training models can effectively improve the models recognition ability. However, the low floating-point operation (FLOPs) model used in the bauxite detection task in this paper cannot provide the same advantage. This phenomenon is called the low FLOPs trap . Because GhostModule uses fewer original convolution operations and a lot of cheap operations to reduce the computational burden, this design provides high computational efficiency, but it may lead to the inability to capture and process the same amount of information as the original convolution operation. In particular, the model may perform poorly when dealing with complex features or fine-grained tasks.

To address the aforementioned issues, this paper replaces the conventional convolution operation in GhostModule with Dynamic Convolution (DynamicConv). This design effectively expands the trainable parameter scale of the model while slightly increasing computational overhead, thereby significantly improving feature representation capabilities. DynamicConv dynamically optimizes the selection and combination strategy of convolution kernels based on the individual characteristics of the input samples, achieving an adaptive feature extraction process. Specifically, the operation mechanism of this module includes two key steps: first, the input features are processed, and a set of convolution kernels W is preset as a set of "experts"; second, the contribution of each "expert" is adjusted through dynamically generated weight coefficients, which are adjusted in real time according to the characteristics of different input samples, as shown in Equation (1).

$$Y = \sum_{i=1}^{M} \alpha_i \left(X * W_k \right) \tag{1}$$

In the formula: Y represents the output feature of dynamic convolution, which is obtained by weighted summation of multiple convolution kernels; Wk is the preset number of convolution kernel groups; X is the input feature map; * indicates the convolution operation. Figure 2 shows the internal structure of DymamicConv. One total convolution kernel is determined by k convolution kernels. The input data is extracted by the attention network to obtain the weights of this convolution kernel, and the weights are linearly weighted to form a new convolution kernel.

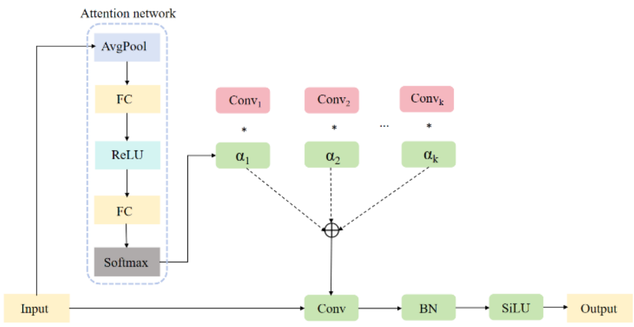

Fig. 2 DynamicConv Structure Diagram

Dynamic convolution, by introducing an adaptive weight generation mechanism, enables the model to flexibly and dynamically adjust the convolution kernels according to input features, significantly enhancing the networks ability to extract different features while maintaining computational efficiency. Based on this idea, this paper embeds the Ghost Dynamic Module into the existing C3k2 structure, thus obtaining the C3k2-GD module.

3.2. LSDECD detection head

The detection head of the YOLOv11 basic model has limitations in bauxite detection, such as high false detection rate for small targets, large number of parameters, and poor performance in fine-grained feature extraction in complex scenes. To address these limitations, this paper designs a lightweight shared detail-enhanced convolutional detection head, LSDECD, to improve detection performance. The head structure is shown in Figure 3.

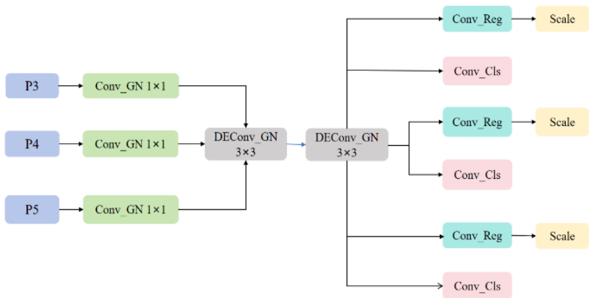

Fig.3 LSDECD module structure diagram

After acquiring the multi-scale feature maps output by the neck network, the LSDECD detection head first performs channel dimension transformation and normalization on the features using 1×1 convolutions. Subsequently, a 3×3 detail enhancement convolution (DEConv) module is used to expand the receptive field. This module not only effectively fuses spatial context information with multi-scale features but also significantly reduces network parameters. In the output stage, classification and regression tasks are performed by two independent 1×1 convolutional layers. The regression branch innovatively incorporates a scale layer with learnable dynamic factors, which automatically adjusts the scale parameters according to different target sizes of the detection head. During training, these parameters are iteratively optimized using gradient descent, enabling the detection box size to adapt adaptively. This architecture design effectively improves detection accuracy while maintaining computational efficiency.

DEConv[7] is a convolution operation designed specifically for image processing tasks, aiming to improve model performance by enhancing local detail features. Its core is differential convolution DC, which captures local detail features of the image by calculating the difference between adjacent pixels. The DEConv network structure is shown in Figure 4. Among them, central difference convolution (CDC) and angular difference convolution (ADC) capture complex local features by calculating the pixel difference in the center or angular direction; horizontal difference convolution (HDC) and vertical difference convolution (VDC) calculate the difference between adjacent pixels in the horizontal or vertical direction, respectively, to enhance edge and contour information. DEConv convolves the input features with 5 parallel convolution kernels and adds the results to obtain the output.

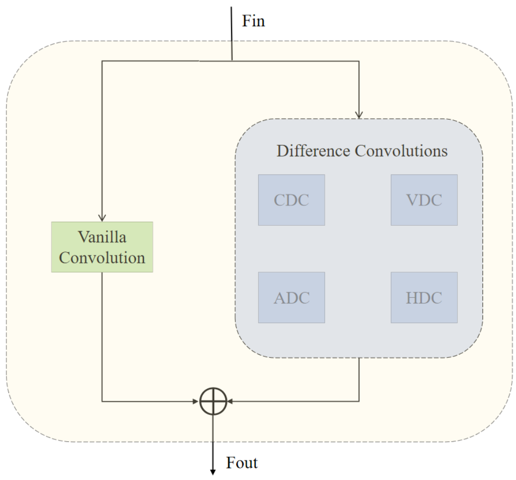

Fig.4 DEConv module structure diagram

4. Experiment and Results Analysis

The dataset for this experiment comes from publicly available online images, combined with bauxite datasets obtained through our own collection. Based on differences in aluminum and iron content, the bauxite is roughly divided into four categories. The training and testing of the bauxite detection and classification algorithm were conducted in the PyCharm integrated development environment; the experimental configuration is shown in Table 1. The algorithm is implemented using the Python language and the PyTorch deep learning framework, and the training process is accelerated using a GPU. All development work was completed under the Windows operating system environment.

Experimental training parameter configuration: The input image size (imgsz) was set to 640 pixels, the training batch size to 32, and the training epoch to 220. The SGD optimizer was used during model training, with an initial learning rate of 0.01 and a learning momentum of 0.937.

Table 1. Experimental Environment Configuration

Configuration Name	Version
Operating System	Windows 11
Programming Language	Python 3.10.14
Deep Learning Framework	Pytorch 2.2.2
GPU	NVIDIA GeForce RTX 3090
CUDA	12.1

4.1. Evaluation Indicators

This paper uses F1 score and mean average accuracy (mAP@0.5) to measure the detection accuracy of the improved model; it uses the number of parameters, floating-point computation (FLOPs), and model size to measure the lightweight performance of the improved model; and it uses FPS to measure the real-time performance of the model.

(1) Precision: The proportion of samples that the model predicts to be positive but are actually positive. Its calculation formula is shown in equation (2).

$$P = \frac{TP}{TP + FP} \tag{2}$$

(2) Recall: The proportion of positive samples that the model predicts as positive samples. Its calculation formula is shown in equation (3).

$$R = \frac{TP}{TP + FN} \tag{3}$$

(3) F1-score: A comprehensive index for evaluating the detection accuracy of the model. It is the harmonic mean of accuracy and recall, taking into account both false negatives and false positives. The higher the F1-score, the better the model is at maintaining high accuracy while also having good recall. The calculation formula is shown in equation (4).

$$F1 = \frac{2PR}{P + R} \tag{4}$$

(4) mAP (mean average precision): mAP refers to the average value of the precision (AP) of all classes. It is obtained by calculating the area under the precision-recall curve for each class. This index represents the average precision at different values. The value of represents the average of all classes. @0.5 represents the average precision value when the IoU threshold between the predicted box and the ground truth box is set to 0.5. The closer this index is to 1, the better the model performance. The calculation formulas for AP and mAP are shown in equations (5) and (6).

$$AP = \int_0^1 P(R)dR \tag{5}$$

$$mAP = \frac{\sum_{i=1}^{K} AP_i}{K} \tag{6}$$

In the formula, K represents the number of categories.

(5) FPS (Frames Per Second): Evaluates the models detection speed, i.e., the frame rate per second, representing the number of images that can be detected per second. In YOLO, calculating FPS requires knowing the image preprocessing time, inference speed, and postprocessing time.

(6) The number of parameters and the amount of computation are important indicators for measuring the complexity and efficiency of deep learning models. They are usually used to evaluate the models storage requirements, computational overhead, and running efficiency on different devices.

4.2. Ablation Experiment

To verify the effectiveness of each improved module, each module was first tested individually to ensure that the selected modules all showed varying degrees of improvement compared to the YOLOv11 baseline model. Then, the C3k2-GD module and the LSDECD module were sequentially superimposed on the YOLOv11 baseline model, and their impact on detection performance was compared and analyzed. The experimental results are shown in Table 2. Where C represents C3k2-GD and L represents LSDECD.

Table 2. Ablation Experiment

Model	FLOPs/G	Params/M	FPS	Precision/%	Recall/%	F1/%	mAP@0.5/%
Base	7.3	3.16	169.66	89.36	88.12	88.64	93.75
C	5.9	2.41	164.10	89.65	89.98	89.78	94.09
L	6.0	2.26	199.23	87.33	86.64	86.93	93.35
C+L	5.6	2.09	185.00	89.77	87.80	88.73	93.60

4.3. Comparative Experiment

Table 3. Comparative Experiment

Model	FLOPs/G	Params/M	FPS	F1/%	mAP@0.5/%
YOLOv5n	6.9	2.13	190.15	86.85	91.22
YOLOv7-tiny	15.8	6.12	145.12	87.34	92.31
YOLOv8n	8.7	3.20	165.24	89.21	93.85
YOLOv10n	8.5	2.98	160.63	88.75	93.74
YOLOv11n	7.3	3.16	169.66	88.64	93.75
CL-YOLOv11	5.6	2.09	185.00	88.73	93.60

To objectively reflect the performance of the improved YOLOv11 algorithm and further verify its effectiveness and generalization, it was compared with six mainstream object detection algorithms (YOLOv3, YOLOv4-tiny, YOLOv5n, YOLOv7-tiny, YOLOv8n, and YOLOv11n) on the same dataset. The experimental results are shown in Table 3.

5. Conclusion

This paper focuses on bauxite detection and sorting, systematically studying and designing an improved target detection algorithm based on YOLOv11. YOLOv11 is improved at both the backbone network and the detection head levels, proposing the C3k2-GD module and the LSDECD lightweight detection head, respectively. Finally, ablation experiments are used to verify the effectiveness of each improved module. Experimental results show that the improved model significantly reduces the number of parameters and computational cost while effectively improving both detection accuracy and real-time performance.

References

[1] Zhen J,Xie B. Fused attention mechanism-based ore sorting network[J]. arXiv preprint arXiv:2405.02785,2024.

[2] Attallah Y,Zigh E,Adda A P. Optimized 3D-2D CNN for automatic mineral classification in hyperspectral images[J]. Reports on Geodesy and Geoinformatics, 2024,118.

[3] Choros K A,Job A T,Edgar M L,et al. Can hyperspectral imaging and neural network classification Be used for ore grade discrimination at the point of excavation? [J]. Sensors, 2022, 22(7): 2687.

[4] Alkhatib M Q. Hyperspectral Image Classification using Spectral-Spatial Mixer Network[J]. arXiv preprint arXiv:2511.15692,2025.

[5] Han K,Wang Y, Guo J, et al. ParameterNet: parameters are all you need for large-scale visual pretraining of mobile networks[C]//Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2024: 15751-15761.

[6] Chen C, Guo Z, Zeng H, et al. Repghost: A hardware-efficient ghost module via re-parameterization[J]. arXiv preprint arXiv:2211.06088, 2022.

[7] Chen Z, He Z, Lu Z M. DEA-Net: Single image dehazing based on detail-enhanced convolution and content-guided attention[J]. IEEE transactions on image processing, 2024, 33: 1002-1015.