Facial Expression Detection Algorithm Based on YOLOv8

Zhengjun Wang
University of Shanghai for Science and Technology, Shanghai, China,

Abstract: Facial expression detection is pivotal for the development of affective computing and human-computer interaction, but existing algorithms often fall short in real-time performance, accuracy, and complexity. This paper presents YOLOv8CRGD, a lightweight facial expression detection algorithm based on YOLOv8. The algorithm features a lightweight cross-scale feature fusion module (CCFM) to enhance the model's adaptability to scale variations and a SENetv2 module to improve feature representation, thereby increasing detection accuracy. To address efficiency in complex visual tasks, an improved visual Transformer structure with a dynamic attention mechanism (DAT) is adopted. Furthermore, GhostConv replaces traditional convolution operations to achieve a lightweight model design. Experimental results show that YOLOv8CRGD achieves a 91.3% accuracy and an mAP50 of 94.4% in facial expression detection tasks, while reducing parameters by 15% compared to the YOLOv8n model. With a frame rate (FPS) of 57.9, the algorithm not only maintains high detection accuracy but also excels in real-time performance, making it a compelling candidate for real-time facial expression analysis applications.

Keywords: YOLOv8; Facial Expression Detection; Lightweight; Dynamic Attention Mechanism; Cross-Scale Feature Fusion

1. Introduction

In recent years, with the rapid development of artificial intelligence and deep learning technologies, facial expression detection has found widespread applications in various fields, including affective computing, human-computer interaction, intelligent surveillance, and psychological analysis. As an important form of non-verbal communication, facial expressions contain rich emotional and intentional information. Therefore, accurately and efficiently recognizing and classifying facial expressions has become one of the research hotspots in the field of computer vision. Currently, facial expression detection algorithms based on deep learning, especially those utilizing Convolutional Neural Networks (CNNs), have significantly improved detection accuracy. However, despite substantial progress in many areas, facial expression detection technology still faces several challenges in practical applications[1].

First, the real-time performance of facial expression detection remains a major obstacle to its widespread application. In real-world scenarios, facial expression detection typically involves processing large amounts of video frame data, which places high demands on algorithmic computational efficiency. Second, the accuracy of facial expression detection algorithms remains inadequate, especially in detecting expressions in complex backgrounds and at different scales. Subtle facial expression changes, individual facial differences, and varied camera angles all impact model accuracy. In particular, in cases of expression blurring, occlusion, and lighting variations, many facial expression detection models exhibit poor robustness. Additionally, due to the multi-scale nature of facial expressions, maintaining consistent accuracy across different scales remains challenging, posing higher requirements for the model's generalization capability.

To address the issues of poor real-time performance, insufficient accuracy, and high model complexity in facial expression detection technology, this paper proposes a lightweight facial expression detection algorithm based on YOLOv8, named YOLOv8CRGD, aiming to achieve higher real-time performance, improved detection accuracy, and lower computational costs[2].

2. Model Introduction

2.1 Yolov8

YOLOv8 (You Only Look Once version 8) is the latest version of the YOLO series object detection algorithms. The YOLO series models are widely used in the field of computer vision due to their excellent real-time performance and efficient object detection capabilities. YOLOv8 inherits and expands upon the technical advantages of YOLOv7 while introducing optimizations in several aspects, leading to further improvements in accuracy, inference speed, and model complexity[3].

2.2 CCFM

The core idea of the lightweight Cross-Scale Feature Fusion Module (CCFM) is to enhance the model's ability to detect multi-scale objects by merging feature maps of different scales. In CCFM[4], low-level feature maps undergo convolution operations to further extract features, while high-level feature maps are upsampled to match the resolution of the low-level feature maps. These feature maps are then merged through concatenation. The upsampling ensures that the high-level semantic features, while preserving their global information, are aligned with the low-level detailed features, resulting in a feature map that combines both detailed and global semantic information.

This text utilizes the output features of the last three stages of the CCFM backbone network as the input to the encoder. The hybrid encoder converts multi-scale features into a series of image features through intra-scale interaction and inter-scale fusion. Subsequently, IoU-aware queries are used to select a certain number of image features from the encoder output sequence, serving as the initial object queries for the decoder. Finally, a decoder with a prediction head iteratively optimizes the object queries to generate bounding boxes and confidence scores.

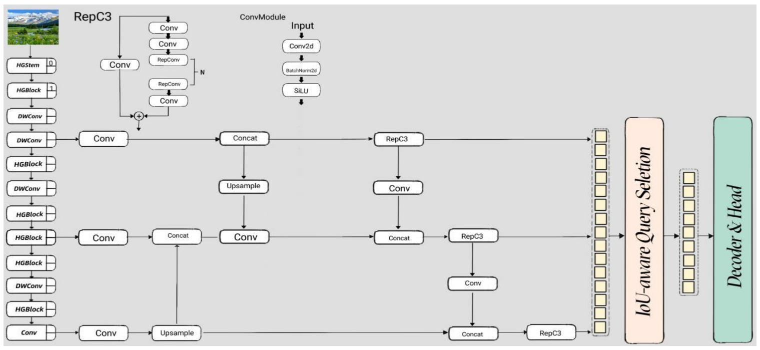

Figure 1. CCFM

2.3 SENetv2

SENetV2 (Squeeze-and-Excitation Networks V2) is an improved convolutional neural network module designed to enhance the global representation capability of neural networks by introducing a more powerful channel attention mechanism[5].

The implementation of the SENetv2 model encompasses critical hyperparameters that dictate its operational dynamics. The channel hyperparameter is essential, defining the number of input feature map channels. This parameter is pivotal as it directly influences the input dimensions for subsequent fully connected layers within the model. The reduction hyperparameter is set to a default value of 16, which determines the dimensionality reduction ratio. This reduction mechanism is crucial for managing the computational load and maintaining the model's efficiency by compressing each channel's feature to one-sixteenth of its original dimension before mapping it back to the original dimension.

The SENetv2 architecture is predicated on the concept of attention mechanisms to enhance the network's responsiveness to salient features while suppressing less relevant ones. The process begins with a Squeeze operation, where global average pooling (AdaptiveAvgPool2d) is applied to condense spatial information of each channel into a single value, resulting in a one-dimensional feature vector. This vector is then subjected to an Excitation operation through two sequential fully connected layers (Linear). The initial layer reduces the feature dimension, followed by a ReLU activation function, and the subsequent layer maps the features back to their original dimension, capped with a Sigmoid activation function to ensure output weights are bounded between 0 and 1, representing the importance weights of each channel.

The Scale operation subsequently applies these learned channel weights to the original feature maps, amplifying the contribution of significant features and muting the less critical ones. In the SELayerV2, the concept is extended through the cardinality hyperparameter, enabling cross-scale feature integration. The input feature map is partitioned into multiple groups along the channel dimension, and the SE operation is applied independently to each group. The outcomes are concatenated and further merged through a final fully connected layer, facilitating a nuanced feature regulation. This approach allows SENetv2 to effectively capture and emphasize useful features at various scales while suppressing noise and irrelevant information, thereby enhancing the network's performance and generalization capabilities.

It primarily involves the following three steps:

2.3.1 Squeeze Operation

In a convolutional neural network, given an input feature map $\mathbf{X} \in \mathbb{R}^{H \times W \times C}$, H is the height， W is the width， C is the number of channels, SENetV2 first performs "Global Average Pooling" (GAP) to reduce the spatial dimensions. This operation compresses the $H \times W$ feature map into a 1×1 representation, preserving only the global information for each channel. The specific formula is:

$$z_c = \frac{1}{H \times W} \sum_{i=1}^{H} \sum_{j=1}^{W} x_{ijc}$$

z_c represents the global description of the， x_{ijc} represents the pixel value of the c th channel in the input feature map $\mathbf{X}$.Through this operation, we obtain a channel-level representation $\mathbf{z} \in \mathbb{R}^{C}$, where global average pooling condenses the spatial information of each channel into a statistical summary.

2.3.2 Excitation Operation

After the squeeze phase, SENetV2 assigns a weight to each channel based on its global description. This weight reflects the importance of the channel in the global context. SENetV2 introduces a fully connected layer to reduce the number of channels from C to a smaller dimension r, where r is a reduction ratio. The expression is as follows:

$$\mathbf{s} = \sigma(\mathbf{W}_2 \delta(\mathbf{W}_1 \mathbf{z}))$$

$\mathbf{W}_1 \in \mathbb{R}^{r \times C}$ is the weight matrix of the first fully connected layer, which reduces the number of channels from C to r,

$\mathbf{W}_2 \in \mathbb{R}^{C \times r}$ is the weight matrix of the second fully connected layer, which restores the number of channels back to C,

$\delta(\cdot)$ represents the activation function, introducing non-linearity,

$\delta(\cdot)$ denotes the sigmoid function, which maps the result to the range [0, 1], representing the weight of each channel.

Through this process, the model learns the importance weights s $\in \mathbb{R}^{C}$

2.3.3 Channel Reweighting

Next, SENetV2 applies the learned channel weights $\mathbf{s}$ to each channel of the input feature map $\mathbf{X}$, thereby emphasizing important features and suppressing less important ones. This operation is performed by multiplying the weights channel-wise, and the formula is as follows:

$$\tilde{\mathbf{X}}_c = \mathbf{X}_c \cdot s_c$$

Where $\mathbf{X}_c$ represents the reweighted output of the cth channel,and s_c is the learned weight for the c th channel.

2.4 DATAttention

DAT (Deformable Attention Transformer) is a variant of the attention mechanism, primarily designed to improve the efficiency and flexibility of traditional Multi-head Self-Attention (MHSA) by introducing a deformable attention mechanism. The key feature of DAT is the use of flexible offsets to adjust the positions of sampled features, enabling it to better adapt to the shapes and positions of different objects. This enhances the model's ability to handle complex visual tasks.

The network structure of DAT is as follows Figure 2:

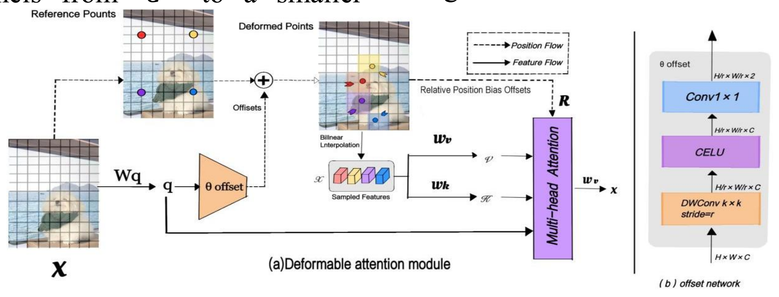

Figure 2. DAT

The working principle of DAT (Deformable Attention Transformer) is as follows:

Deformable Attention Module

The core idea of the deformable attention module is to generate a set of offsets from each query point and use these offsets to sample features from the input feature map. Each query point q corresponds to an initial reference point. These reference points are uniformly distributed across the feature map, marking the initial position of each query point. By learning an offset θ_{offset}, these reference points are dynamically adjusted. The offset is learned by the network through convolution or fully connected layers. Sampling from the feature map is then performed based on these adjusted reference points. The sampling process can be implemented using techniques such as bilinear interpolation.

The mathematical expression is:

$$x' = \text{BilinearInterpolation}(x, \theta_{\text{offset}}(q))$$

Here, x' represents the sampled feature, x is the input feature map, and $\theta_{\text{offset}}(q)$ is the learned offset.

By combining traditional multi-head self-attention with deformable attention, DAT enables more precise capture and localization of complex objects. This not only enhances the model's flexibility in handling variations in shape and position but also improves computational efficiency. The introduction of deformable attention allows DAT to exhibit greater adaptability and performance in visual tasks.

2.5 GhostConv

GhostConv is a lightweight convolution operation designed to reduce redundant computations in convolutional neural networks. The basic idea of GhostConv is to generate a portion of the "base" feature maps using a small number of convolution operations, and then produce additional feature maps through simple linear transformations (such as pointwise convolutions), forming the final output. This method avoids the need to compute all feature maps through traditional convolution operations, thereby significantly reducing computational cost.

Assuming the input feature is X, the standard convolution can be expressed as:

$$Y = X * W$$

Here, W represents the convolution kernel, and $*$ denotes the convolution operation. For GhostConv, the process is completed in two steps:

1. Generate base features:

$$Y_1 = X * W_1$$

2. Generate Ghost features:

$$Y_2 = g(Y_1) * W_2$$

Where $g(\cdot)$ represents a simple transformation operation, such as pointwise convolution.

The final output is:

$$Y = [Y_1, Y_2]$$

$[Y_1, Y_2]$ denotes the concatenation of the base features and the Ghost features.

3. Experimental Results and Analysis

3.1 Dataset

The custom dataset utilized in this study integrates publicly available facial expression datasets from Kaggle with facial expression images collected and labeled from the internet. Special attention was given to data cleaning, the accuracy of labeling, and the balance of sample distribution during the construction of the dataset to ensure its representativeness and accuracy. To ensure data quality, strict quality control was applied to all images, excluding those with low resolution, blurred, or irrelevant to the research task. Duplicate images were removed, and image standardization was performed to reduce biases from different data sources. The dataset comprises 2,156 facial expression images, divided into training and testing sets in a 5:1 ratio. The expressions are categorized into five classes: happiness, sadness, surprise, fear, and anger. To ensure a balanced sample distribution, we performed oversampling on minority classes and undersampling on majority classes. Furthermore, we focused on collecting facial expression images from individuals of different ages, genders, and ethnicities to enhance the diversity and inclusiveness of the dataset.

3.2 Model Training

The hardware and software configuration used in this paper includes:

- 13th Gen Intel(R) Core(TM) i9-13900HX, 2200 MHz,
- GTX 3090Ti 4GB GPU,
- Windows 11 operating system,
- PyTorch 1.7.1 deep learning framework.

3.3 Evaluation Metrics

The performance of the algorithm is evaluated using the following metrics:

- mAP (mean Average Precision): A key metric for assessing overall model performance in object detection tasks.
- Precision: The accuracy of the model's predictions.
- FPS (Frames Per Second): The real-time performance of the model, indicating how many frames the model can process per second.
- Params: The total number of trainable parameters in the model, used to assess the model's complexity.

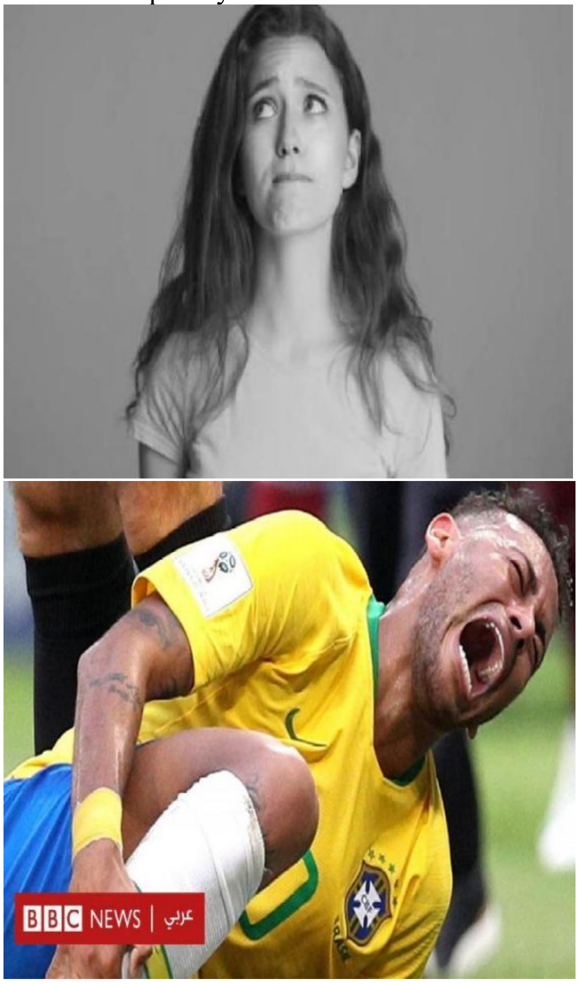

Figure 3. Shown

3.3.1 mAP (mean Average Precision)

mAP (mean Average Precision) is a key metric for evaluating the overall performance of a model in object detection tasks, assessing the detection accuracy across all categories.

Precision-Recall Curve: For each category, the model's detection results are sorted by confidence scores from high to low, and then Precision and Recall are calculated.

Precision (P): This indicates the proportion of true positives among the predicted positive samples.

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

Where TP refers to True Positives, and FP refers to False Positives.

Recall (R): Recall represents the proportion of actual positive cases that are correctly identified as positive by the model.

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

FN is False Negative.

AP (Average Precision): For each category, based on different confidence thresholds, a Precision-Recall curve is plotted. AP is the area under this curve, which can be calculated through integration:

$$AP = \int_0^1 P(R)\, dR$$

mAP (mean Average Precision): mAP is the average of the AP values across all categories:

$$mAP = \frac{1}{n}\sum_{i=1}^{n} AP_i$$

n is the number of categories.

3.3.2 Precision (Accuracy)

Precision measures the accuracy of the model's detection results. It represents the proportion of true positive samples among the predicted positive samples.

The formula for calculating precision is:

$$Precision = \frac{TP}{TP+FP}$$

TP: The number of true positive samples correctly detected by the model.

FP: The number of false positive samples, where the model incorrectly detects negative samples as positive.

3.3.3 FPS (Frames Per Second)

FPS is a key indicator of the algorithm's real-time performance, representing how many frames the model can process per second.

The calculation method for FPS is:

$$FPS = \frac{N}{T}$$

where N is the number of frames, and T is the time taken to process these N frames (in seconds).

3.3.4 Params

Params refers to the total number of trainable parameters in the model and is used to measure the model's complexity. A larger number of parameters typically indicates a more complex model, requiring more computational resources.

For a convolutional layer, the formula to calculate the number of parameters is as follows:

$$\mathrm{Params} = (k \times k \times C_{in} \times C_{out}) + C_{out}$$

where:

- $k \times k$ is the size of the convolution kernel,
- C_{in} is the number of input channels,
- $C_{out}$is the number of output channels,
- C_{out} is also the number of bias terms (one for each output channel).

For a fully connected (dense) layer, the formula to calculate the number of parameters is:

$$\mathrm{Params} = (N_{in} \times N_{out}) + N_{out}$$

where:

- N_{in} is the number of input neurons,
- N_{out} is the number of output neurons.

These metrics combined—precision, speed (FPS), and complexity (Params)—offer a comprehensive evaluation of the YOLOv8CRGD model's performance in terms of accuracy, processing speed, and model complexity.

3.4 Ablation Study

Table 1. Ablation Study

	mAP@0.5/%	Precision/%	FPS
Yolov8n	89.2	87.4	59.6
+CCFM	94.5	90.3	58.4
+RSNetv2	93.6	90.6	54.5
+Ghostconv	94.1	88.2	58.9
+DATAttention	94.7	91.1	57.6

By conducting ablation experiments on YOLOv8n, the impact of different modules on model performance is clearly demonstrated. The experimental results are shown in the table. The introduction of the CCFM module increased the mAP from 89.2% to 94.5%, and Precision improved from 87.4% to 90.3%. This improvement in accuracy validates the effectiveness of the CCFM module, which enhances detection accuracy, as shown in table 1.

Although the RSNetv2 module also improved the mAP (from 89.2% to 93.6%), it had a significant impact on inference speed, reducing FPS from 59.6 to 54.5. This indicates that while the RSNetv2 module greatly improves accuracy, its complexity increases computational costs. This may be due to the more complex multi-scale convolution operations performed during feature extraction, adding computational overhead.

In contrast, the GhostConv module achieved a more balanced performance improvement. After introducing GhostConv, the model's mAP reached 94.1%, and the number of parameters decreased from 3.0M to 2.7M, demonstrating that GhostConv reduces redundant computations and enhances model efficiency.

The addition of the DATAttention module significantly improved the performance of the YOLOv8n model. By effectively enhancing the model's ability to capture global contextual information, it greatly boosted detection performance.

3.5 Comparative Experiments

Table 2. Comparative Experiments

	mAP@0.5/%	Precision/%	FPS	Params/M
Faster RCNN	82.1	81.4	29.4	15.0
Yolov5	90.3	86.9	45.3	2.5
Yolov7	88.7	89.5	48.8	2.7
SSD	82.9	82.5	25.2	17.4
CenterNet	90.5	86.1	50.4	6.4
Yolov8+CBMA	91.7	90.2	55.7	3.1
Yolov8n	89.2	87.4	59.6	3.0
Yolov8-CRGD	94.4	91.3	57.9	2.8

In the comparative experiments, several classical object detection algorithms were comprehensively compared with the improved YOLOv8 series, evaluating their performance in terms of accuracy, speed, and parameter count. The experimental results highlight the strengths and weaknesses of different algorithms in practical applications as shown in table 2.

The Faster RCNN model, as a region proposal-based object detection method, shows significantly lower inference speed compared to other lightweight models. This can be attributed to the region proposal and feature resampling steps involved in the detection phase, which increase computational complexity. Therefore, it is not well-suited for applications with high real-time requirements.

For the SSD model, despite its widespread use in early object detection tasks, the experimental results show that SSD's mAP and Precision are lower than those of more modern object detection models, making it less suitable for real-time applications in contemporary scenarios.

The CenterNet model demonstrates a good balance in performance, falling into a moderate range. Its center point prediction method reduces the computational cost compared to traditional two-stage models while maintaining relatively good detection accuracy. However, it still lags behind the latest versions of the YOLO series in extreme real-time and high-precision tasks.

Finally, the proposed YOLOv8 series models achieve an exceptional balance between accuracy and speed. This shows that with the introduction of the efficient CRGD module, the YOLOv8 series not only reaches leading accuracy levels but also maintains the efficiency of its lightweight structure, achieving high detection accuracy and real-time performance.

3.6 Visualization Analysis

Below are samples from the facial expression detection tests. It can be seen that the model demonstrates high accuracy in detecting facial expressions, accurately identifying facial features and expressions with considerable precision.

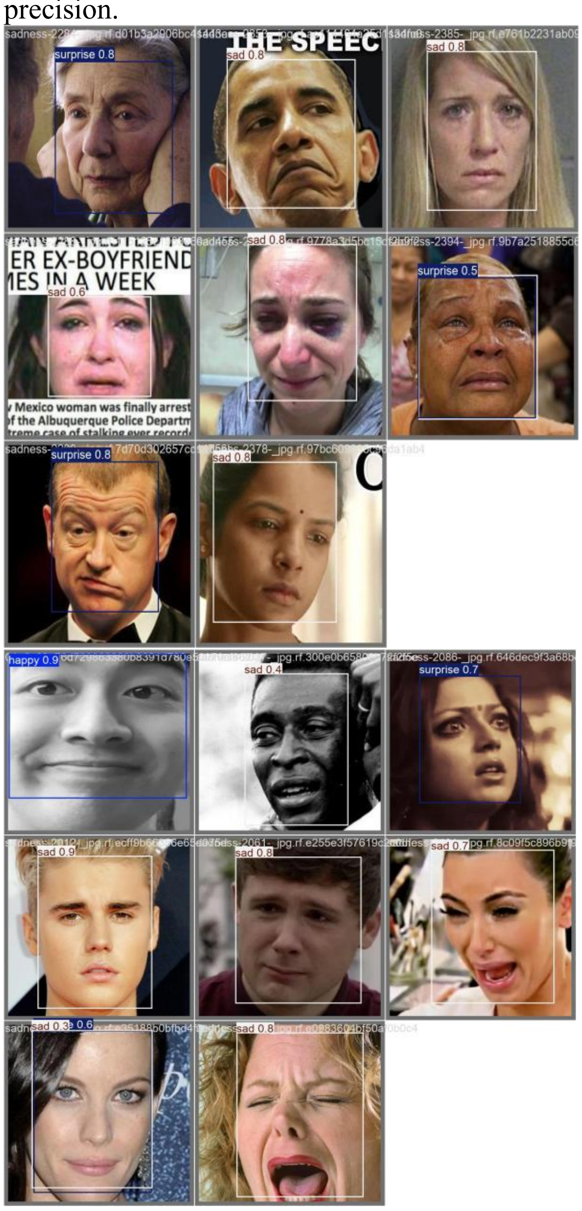

Figure 4. Detection

4. Conclusion

In this paper, we have applied various module enhancements to the YOLOv8n model and conducted comparative experiments with several classical object detection algorithms to explore the performance of different models and modules in terms of detection accuracy, inference speed, and parameter complexity. The ablation study revealed that the integration of modules such as the Cross-Scale Feature Fusion Module (CCFM), SENetv2, GhostConv, and Dynamic Attention (DAT) significantly improved the model's mean Average Precision (mAP) and Precision, with CCFM and DAT showing the most substantial effects.

The CCFM facilitates the fusion of multi-scale features, enhancing the model's ability to detect objects of varying sizes by integrating information from different spatial resolutions. This theoretical underpinning is rooted in the concept that combining features at various scales can capture both local and global context, leading to a more comprehensive representation for object detection tasks.

The SENetv2 module, which employs a sophisticated channel attention mechanism, refines feature representation by emphasizing informative features and suppressing less useful ones. This is based on the theoretical framework that not all features contribute equally to the detection task, and by selectively focusing on the most salient features, the model can achieve higher accuracy.

GhostConv, which replaces traditional convolutions, reduces the computational burden while maintaining accuracy. This is achieved by factorizing the convolution operation into a depthwise convolution followed by a pointwise convolution, which theoretically allows for a more efficient use of parameters without sacrificing detection performance.

The DAT module, based on the Transformer architecture, introduces a dynamic attention mechanism that adapts to the complexity of the visual input, theoretically allowing the model to focus on the most relevant parts of the input for detection tasks, thus improving efficiency and accuracy.

Although some modules marginally affected inference speed, the overall performance saw significant improvements. Notably, GhostConv not only enhanced accuracy but also substantially reduced the number of parameters, making it particularly suitable for scenarios with limited computational resources.

In comparative experiments, the YOLOv8 series models outperformed traditional detection models such as Faster RCNN, SSD, and CenterNet in both detection accuracy and real-time performance. The YOLOv8-CRGD model, while maintaining a small parameter count, achieved a mAP of 94.4% and a Precision of over 91%, with an inference speed of 57.9 FPS. This demonstrates that our model achieves an excellent balance between efficiency and accuracy, making it a strong candidate for real-time facial expression analysis applications.

References

[1] Hu Haixia, Xiang Cenyang, Fan Weiqin. Research on Tray Recognition Method Based on Improved YOLOv8n Model [J/OL]. Journal of Chongqing Technology and Business University (Natural Science Edition), 1-11 [2400-09-23] http://kns.cnki.net/kcms/detail/50.1155.N.20240914.1645.007.html.

[2] Hu Jiusong, Liu Zhangchi, Yu Qian, et al. YOLOv8 Smoke Recognition Algorithm Incorporating GhostNet and CBAM [J/OL]. Journal of Electronic Measurement and Instrumentation, 1-8 [2400-09-23] http://kns.cnki.net/kcms/detail/11.2488.TN.20240906.1038.003.html.

[3] Wang Baocheng, Yuan Hao, Han Feng, et al. Track surface damage detection method based on deep vision algorithm [J/OL]. Experimental Technology and Management, 1-12 [2400-09-23] http://kns.cnki.net/kcms/detail/11.2034.T.20240903.0944.004.html.

[4] Cheng Shun, Li Jianrong, Wang Zhiqian, et al. A lightweight underwater optical image recognition algorithm based on YOLOv8 [J/OL]. Advances in Laser and Optoelectronics, 1-20 [2022-09-23] http://kns.cnki.net/kcms/detail/31.1690.tn.20240809.1638.010.html.

[5] Yang Feng, Yao Xiaotong. A lightweight model for detecting wheat leaf diseases and pests based on improved YOLOv8 [J]. Smart Agriculture (Chinese and English), 2024, 6 (01): 147-157.