Project Readme

Student: Bachhav Aryan Kishor
Roll Number: 210253
Paper Name Cross-Modality Fusion Transformer for Multispectral Object Detection

Brief Overview

- **Cross-Modality Fusion Transformer (CFT)**: The CFT utilizes the self-attention mechanism of Transformers for intra-modality (within each type of data) and inter-modality (between different types of data) fusion. This improves detection accuracy by integrating RGB and thermal images more effectively.
- **Two-Stream Backbone**: Built on the YOLOv5 framework, the two-stream feature extraction network enhances performance by leveraging the complementary nature of RGB and thermal modalities.

New Idea /Function Added

1. **Replace `Conv` with CrossConv:**
   - **Reason:** CrossConvolution **(CrossConv)** allows for more effective feature interaction between channels or modalities (such as RGB and thermal data) by enabling cross-channel combinations. This improves feature fusion and extraction, give better performance compared to standard convolution, specifically in multimodal data .
2. **Replace `SPP` with `SPPF`:**
   - **Reason:** The `SPPF` layer give improved pooling , resulting in better feature extraction. This can be advantageous for preserving spatial hierarchies and retaining more information from the feature maps.

Standard Convolution (Conv):
The computational cost of a standard convolutional layer is:

$$Cost_{Conv} = k_h \times k_w \times C_{in} \times C_{out} \times H \times W$$

Where: - k_h, k_w are the kernel dimensions, - C_{in}, C_{out} are the input and output channels, - H, W are the spatial dimensions of the input feature map.
CrossConvolution (CrossConv):
The computational cost for CrossConv is:

$$Cost_{CrossConv} = \frac{k_h \times k_w \times C_{in} \times C_{out} \times H \times W}{G}$$

Where G is the number of groups for cross-channel interaction ($G \leq C_{in}$).
For $G = 2$, the computational cost becomes:

$$Cost_{CrossConv} = \frac{1}{2} \times Cost_{Conv}$$

Result : So, CrossConv reduces computation by a factor of $\frac{1}{G}$

Table 1: Comparison after 50 epoch

	Before Enhancement	After Enhancement
Recall	90.9	92.7
Prescision	93.3	94.7
mAP 50	94.5	96.3
mAP 75	59.8	66.3
mAP	57.9	58.2

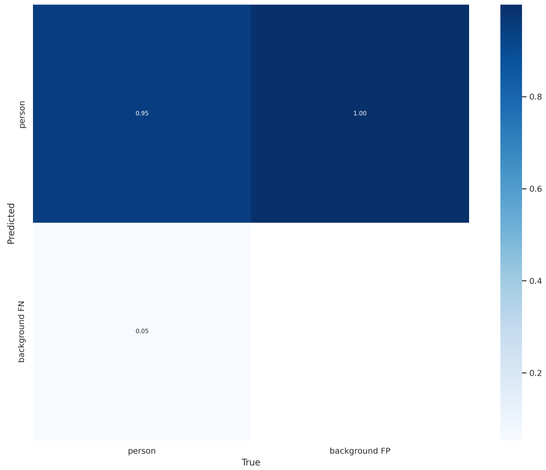

Figure 1: Confusion Matrix Part2

Standard SPP (Spatial Pyramid Pooling):
The computational cost for SPP is:

$$Cost_{SPP} = \sum_{i=1}^{n} P_i \times C \times H \times W$$

Where: - P_i is the pooling size at level i, - C is the number of channels, - H, W are the spatial dimensions. the cost is lineraly proportinal to n .
SPPF (Spatial Pyramid Pooling - Fast):
SPPF reduces this by applying a single pooling operation and reusing results:

$$Cost_{SPPF} = C \times H \times W$$

Result: SPPF reduces the cost from $O(n \times P \times C \times H \times W)$ to $O(C \times H \times W)$, providing faster performance with similar feature quality.

1 Comparison

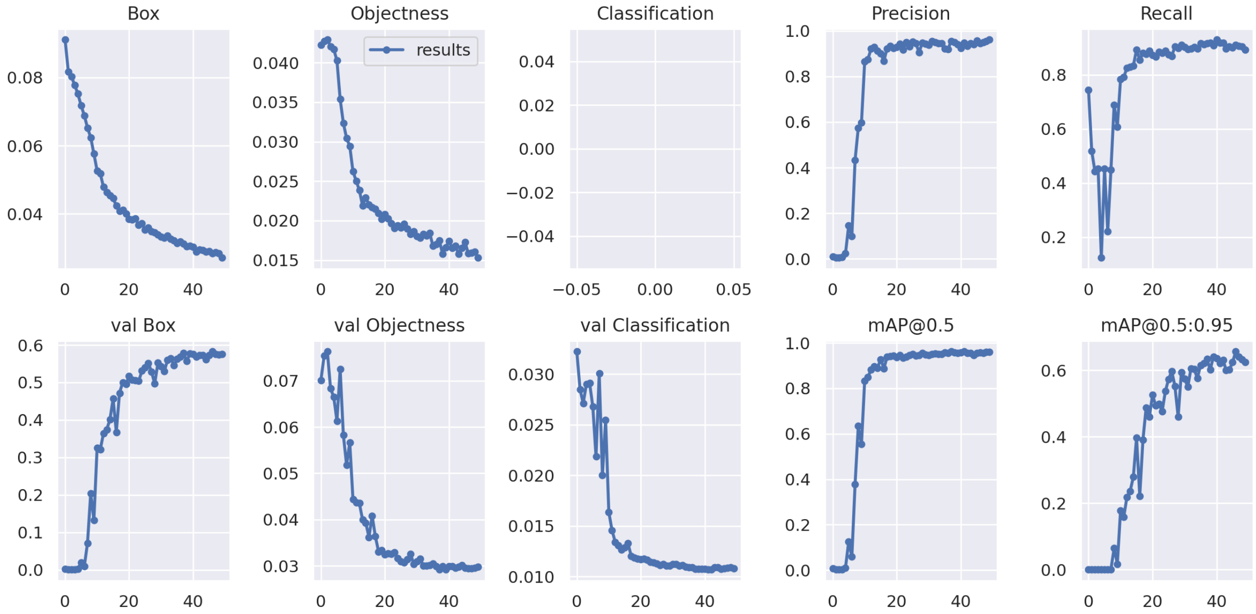

Figure 2: Result$_{part2}$

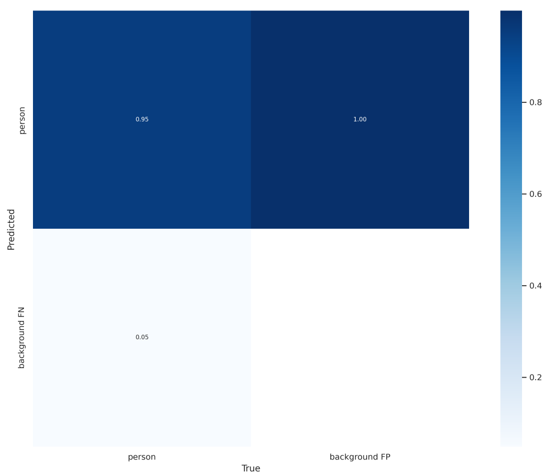

Figure 3: Confusion matrix part1

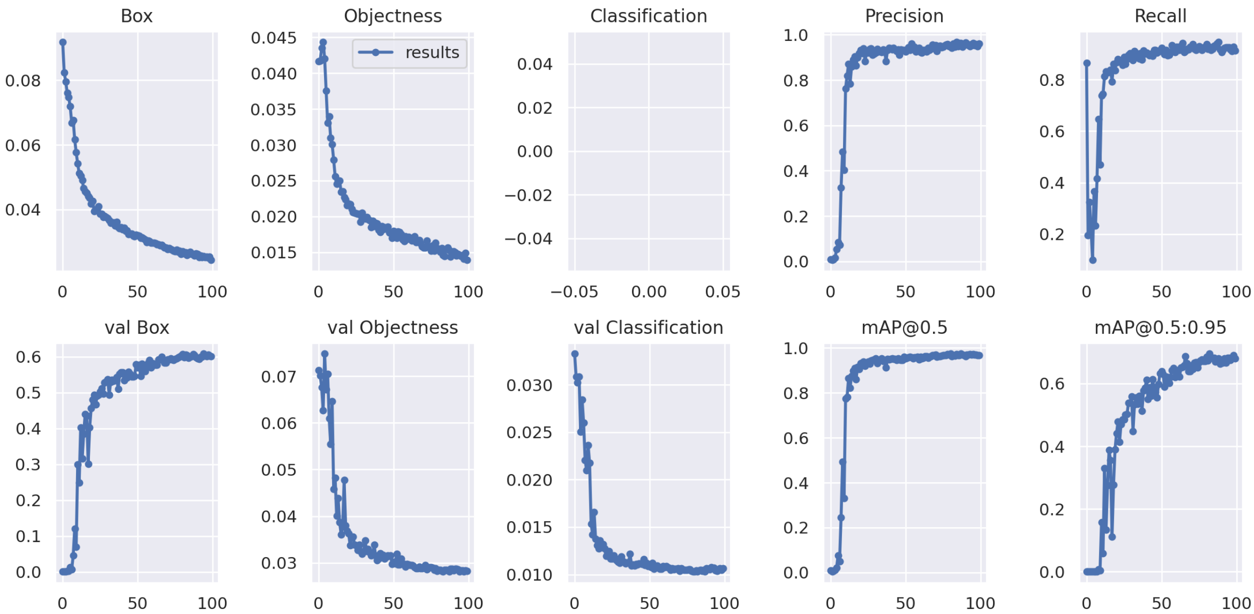

Figure 4: Result Par1