Pedestrian Detection in Adverse Conditions via Cross-Modal Fusion and a YOLO-Style Head

Kerem Erciyes

Department of Artificial Intelligence for Science and Technology, University of Milan

Abstract :

Object detection in different lighting conditions remains a significant challenge for autonomous systems. Standard RGB cameras fail in low-light scenarios, while infrared (IR) sensors, though effective at thermal detection, often lack rich semantic detail. This project presents a deep learning pipeline that leverages the complementary strengths of paired RGB and IR image pairs for robust object detection. The proposed model integrates a dual-stream ResNet backbone for parallel feature extraction, a bidirectional cross-attention mechanism and axial attention mechanism for multi-modal fusion, and a lightweight YOLO-style head for efficient, single-stage object detection. The system is trained and evaluated on the LLVIP dataset. This report details the model's architecture, the anchor-based loss computation, and the refined training methodology. A significant portion of the investigation focuses on addressing training instabilities, particularly the gradient explosion encountered when unfreezing the backbone. This was tried to be managed through systematic hyperparameter tuning, a differential gradient clipping strategy, and the implementation of a cosine learning rate scheduler with a warmup phase, providing critical insights into the challenges of fine-tuning complex fusion models.

Index Terms: Object Detection, Multi-Modal Fusion, Deep Learning, Computer Vision, YOLO, Infrared, Low-Light Vision, Cross-Attention, Training Stability, SGD.

1 Introduction

The ability to reliably detect objects, particularly people, is a cornerstone of many applications, from autonomous driving to public safety surveillance. However, the performance of standard object detection models, which typically rely on visible-spectrum (RGB) imagery, degrades significantly in low-light or nighttime scenarios. Infrared (IR) cameras, which capture thermal radiation, offer a powerful alternative as they are invariant to illumination. While IR can effectively highlight warm objects like people against a cold background, it lacks the rich color and texture information present in RGB images.

This project addresses the challenge of low-light pedestrian detection by combining the strengths of both RGB and IR sensors. The core hypothesis is that by effectively combining features by fusion from these two complementary modalities, a model can achieve more robust and accurate detection performance than by using either spectrum alone.

The architecture centered around two key components: a fusion strategy and an efficient detection head. The model utilizes parallel ResNet-18 backbones to extract features from paired RGB and IR images. These features are then progressively combined using a custom **Bidirectional Cross-Attention** module, which allows the two data streams to query each other and exchange information. The fused feature map is then fed into a **YOLO-style detection head**, which treats object detection as a regression and classification problem on a grid of anchor boxes. This single-stage approach is computationally efficient and, when combined with post-processing step, Non-Maximum Suppression (NMS), provides better detection results. This paper gives more detail of the architecture of this model, the dataset used for training, the specific loss functions and training procedures, and then discusses the results and potential future works.

2 Dataset and Preprocessing

This project utilizes the LLVIP dataset, a public benchmark specifically designed for low-light vision tasks.

2.1 The LLVIP Dataset

The `LLVIP` dataset consists of 15488 of paired visible RGB and infrared images captured in various low-light surveillance scenarios. The primary object class is 'person'. The dataset is structured with predefined training and test splits, providing a basis for training and evaluation. The custom `LLVIPDataset` class loads these splits for use in the PyTorch `DataLoader`.

2.2 Preprocessing and Augmentation

To prepare the data for the model and improve its ability to generalize, a series of preprocessing and augmentation steps are applied to the training data:

- **Image Resizing:** All input images (both RGB and IR) are resized to a fixed resolution of 512×512 pixels. This ensures consistent input dimensions for the model's backbone.
- **Data Augmentation:** To prevent overfitting and expose the model to a wider variety of data, several dynamic data augmentation techniques are applied to the training set. These augmentations are on the fly so they are randomly applied to each batch.
  - **Color Jitter:** The brightness, contrast, saturation, and hue of the RGB images are randomly adjusted.
  - **Geometric Transformations:** A `RandomAffine` transformation is applied, introducing rotations of up to 10 degrees, translations, and scaling.
  - **Horizontal Flipping:** With a `50%` probability, both the RGB and IR images in a pair are flipped horizontally. The corresponding bounding box coordinates are adjusted accordingly.
- **Tensor Conversion and Normalization:** Images are con-

verted to PyTorch tensors. The RGB images are normalized using the standard ImageNet mean and standard deviation values. The single-channel IR images are normalized with a mean and standard deviation of 0.5.

3 Methodology and Model Components

The model architecture is a three steps pipeline designed for Feature Extraction, Feature Fusion, and Object Detection by interpreting information from the two input spectra. Figure 1 shows a detailed diagram of the pipeline.

3.1 Backbone: Multi-Scale Feature Extraction

The foundation of the model is a dual-branch backbone architecture. Two separate **ResNet-18** models are employed, sourced from the `timm` library, to act as feature extractors: one for the RGB stream and one for the IR stream. To handle the single-channel IR input, the weights of the first convolutional layer of the IR backbone are adapted by averaging the original three-channel pretrained weights. Both backbones output feature maps from three different stages (`stage2`, `stage3`, and `stage4`), providing multi-scale representations of the input.

3.2 Hybrid Feature Extraction with Progressive Fusion

This is the core of the model's innovation, where information from the two modalities is merged at each feature scale.

- **Bidirectional Cross-Attention Fusion:** This module is the primary fusion mechanism. For a given feature scale, it performs attention in two directions: The RGB features query the the IR features, and the IR features query the the RGB features. This allows each modality to dynamically incorporate the most relevant information from the other.
- **Convolutional Gating:** A convolutional gating mechanism learns a set of four attention weight maps. These weights combine the original and cross-attended features, allowing the model to decide on a pixel-by-pixel basis how much to trust each of the four feature streams, which are: original RGB, original IR, fused RGB, fused IR. The weights are enforced to sum to one at each pixel via a `softmax` operation across the channel dimension.
- **Axial Attention Block (Optional):** This module can be applied after cross-modal fusion. It performs self-attention sequentially along the height and then the width axes of the feature map, efficiently providing self-attention model captures long-range spatial dependencies within the fused representation.
- **ViT Enhancer Block (Optional):** This module is designed to refine the features from each modality before they undergo cross-modal fusion. The purpose of this module is to leverage the strength of Vision Transformers (ViT) in capturing global context, complementing the local feature extraction.

3.3 Supervision Head

To provide a more stable gradient signal to the deep layers of the backbones, **deep supervision** strategy is applied. A small, classification-only convolutional head (`SimpleConvHead`) is attached directly to the highest-level feature map (`layer4`) of one of the backbones before fusion. This head performs dense, per-pixel classification. The resulting auxiliary cross-entropy loss ensures that the backbone is encouraged to learn, mitigating potential vanishing gradient issues.

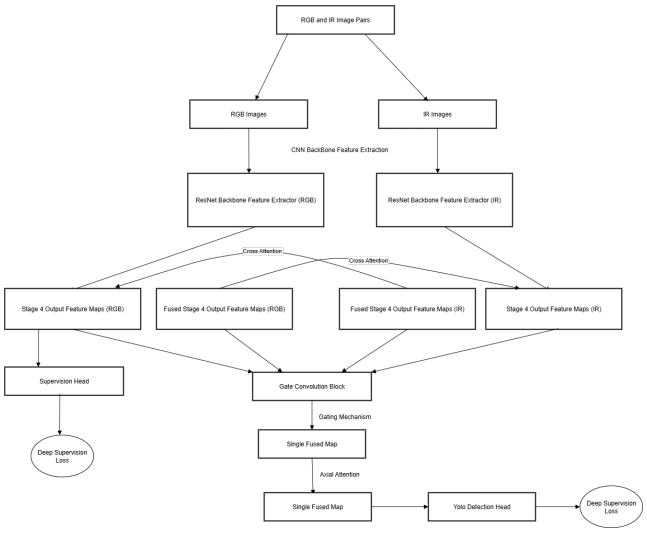

Figure 1. Model Diagram

3.4 YOLOv3-Style Detection Head

The final detection stage uses a `YOLOHead`, a lightweight and efficient single-stage detector head.

- **Anchor-Based Detection:** The head operates on the final fused feature map (e.g., a 16×16 grid). At each grid cell, it predicts offsets for a set of predefined bounding boxes of various shapes, known as **anchor boxes**.
- **Decoupled Prediction:** To improve stability, the head uses three separate 1×1 convolutional layers to predict the three distinct components of a detection:
  1. **Box Regression (`conv_box`):** Predicts the `xywh` coordinate adjustments for each anchor.
  2. **Objectness Score (`conv_obj`):** Predicts the confidence that an anchor box contains an object versus background.
  3. **Class Probability (`conv_cls`):** Predicts the probability of the object belonging to a specific class (in this case, 'person').

4 Loss Computation

The model's learning is guided by a multi-component loss function managed by the `YOLOLoss` class, designed to work in harmony with the YOLO-style head.

4.1 The YOLO Loss Criterion (`YOLOLoss`)

Unlike the classical losses, the YOLO loss first matches ground-truth boxes to anchor boxes based on their shape and location. It then computes a weighted sum of three distinct loss components for the matched pairs:

- **Box Loss (`loss_box`):** A **Complete IoU (CIoU)** loss is used for bounding box regression. CIoU is an advancement over

standard IoU loss as it accounts for the overlap area, central point distance, and aspect ratio consistency between the predicted and ground-truth boxes, leading to faster and more accurate convergence.

- **Objectness Loss (`loss_obj`):** A **Binary Cross-Entropy (BCE)** loss is calculated across all anchors on the feature map. It trains the model to distinguish between foreground (anchors containing objects) and background.
- **Classification Loss (`loss_cls`):** A **BCE** loss is calculated only for anchors that contain an object. It pushes the model to correctly classify the object within the box. **Label smoothing** is applied to this loss to regularize the model and prevent overconfidence.

4.2 Total Loss Combination

The final loss that is backpropagated is a weighted sum of the three YOLO losses and the auxiliary deep supervision loss:

$$\mathcal{L}_{\text{total}} = w_{\text{box}}\mathcal{L}_{\text{CIoU}} + w_{\text{obj}}\mathcal{L}_{\text{BCE\_obj}} + w_{\text{cls}}\mathcal{L}_{\text{BCE\_cls}} + \lambda_{\text{DS}}\mathcal{L}_{\text{DS}}$$

where w represents the weights for each YOLO loss component and λ_{DS} is the weight for the deep supervision loss.

5 Training and Validation

The model was trained and tuned for stability and performance, addressing challenges discovered during experiment.

5.1 Training Procedure

The training loop orchestrates the learning process.

- **Optimizer and Scheduler:** A **Stochastic Gradient Descent (SGD)** optimizer with Nesterov momentum was used, which was found to provide better generalization than Adam-based optimizer. It uses differential learning rates (a lower rate for the backbone, a higher rate for the other layers). This is paired with a **Cosine Annealing learning rate scheduler with a linear warmup phase**. The warmup gradually increases the learning rate over the first epoch, preventing instability, after which the rate cyclically decays.
- **Progressive Unfreezing:** The backbones are initially frozen to allow the new fusion and detection layers to stabilize. After a set number of epochs, the backbone is progressively unfrozen layer by layer.
- **Gradient Management:**
  - **Mixed Precision:** Automatic Mixed Precision (`torch.cuda.amp`) is used to speed up training and reduce memory usage.
  - **Gradient Accumulation:** Gradients are accumulated over several batches before an optimizer step is taken, effectively simulating a larger batch size.
  - **Gradient Clipping:** To prevent the **gradient explosion** observed upon unfreezing the backbone, a differential clipping strategy is employed. Gradients for the sensitive backbone are clipped to a small norm (e.g., 1.0), while gradients for the more robust fusion and detection heads are allowed a larger norm (e.g., 10.0). A temporary, much lower learning rate is also applied for a few epochs immediately after unfreezing to ensure a smooth transition.

5.2 Post-Processing and Inference

During validation and testing, the raw output from the `YOLOHead` must be post-processed to generate the final set of detections. This involves:

1. **Decoding Predictions:** The raw model outputs (offsets, objectness scores, class logits) are converted into absolute bounding box coordinates (`xyxy`) and confidence scores.
2. **Non-Maximum Suppression (NMS):** This crucial step eliminates redundant, overlapping bounding boxes for the same object. It iteratively selects the box with the highest confidence score and removes any other nearby boxes that have a high Intersection over Union (IoU) with it.

6 Experiments and Results

This section details the training configuration and presents an analysis of the model's performance. The quantitative results, including validation loss and mean Average Precision (mAP). The following visualizations provide insight into the model's internal mechanisms and its final detection capabilities on unseen data.

6.1 Training Configuration

The model was trained on a single NVIDIA T4 x2 GPU. The key hyperparameters governing the training process were configured as follows:

- **Optimizer:** SGD with Nesterov momentum (0.937).
- **Scheduler:** Cosine Annealing with a linear warmup over the first epoch.
- **Learning Rates:** Initial rate of 1×10^{-2} for the head/fusion layers and 1×10^{-3} for the backbone.
- **Weight Decay:** 5×10^{-4}.
- **Batch Size:** 8, with 4 steps of gradient accumulation, resulting in an effective batch size of 32.
- **Image Size:** 512×512 pixels.
- **Subset Size:** 5000 image pairs.
- **Epochs:** 100, with an early stopping patience of 15 epochs based on the validation loss and mAP@.50 score.

6.2 Training and Validation Loss

The average training and validation loss per epoch is shown in Figure 2.

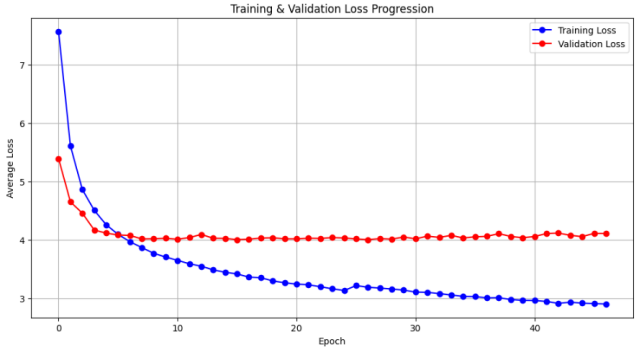

Figure 2. Epoch-Loss Plot

6.2.1 Prediction Examples Figure 3, Figure 4 and Figure 5 shows the model's final detection outputs. The predicted bounding boxes (red) are compared against the ground-truth boxes (green).

Figure 3. Prediction on Training(RGB)

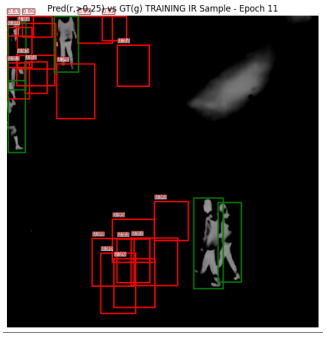

Figure 4. Prediction on Training(Ir)

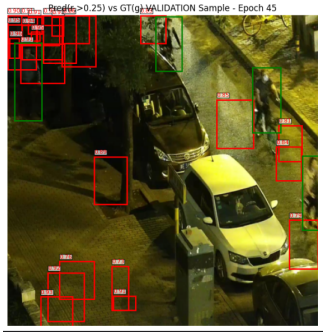

Figure 5. Prediction on Validation

6.2.2 Analysis of Attention Maps Figure 6 visualizes the four spatial weights.

7 Conclusion

This report has detailed the architecture and training of a deep learning model for multispectral pedestrian detection. By leveraging a dual-stream ResNet backbone and a progressive fusion strategy featuring Bidirectional Cross-Attention, the model creates a unified feature representation. This representation is then processed by a YOLO-style detection head. The project's findings are unique architecture and the training methodology. Mostly overcoming initial overfitting and gradient explosion required a systematic approach, with a robust training process that includes

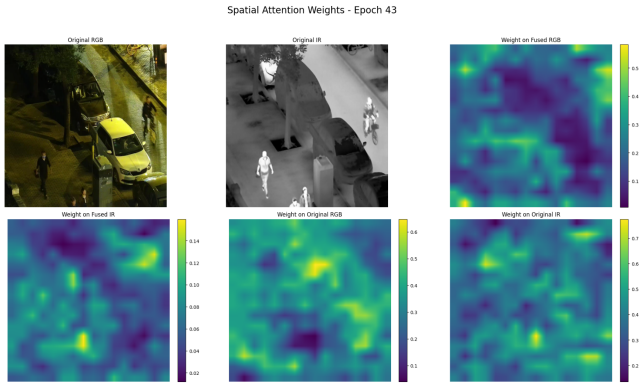

Figure 6. Attention weights

a warmup-decay scheduler, progressive unfreezing with adaptive learning rates, and differential gradient clipping. The final result is a satisfying and effective pipeline that combines the complementary strengths of RGB and IR imagery for the object detection task.

7.1 Discussion and Future Work

Several avenues exist for future work:

For future work, current model's anchor-based YOLO head may struggle with objects of unusual aspect ratios not well-represented by the predefined anchors. Firstly, focusing on maximizing the existing architecture would be better. Activating the optional TimmViTEnhancer to enrich features pre-fusion and implementing a Feature Pyramid Network (FPN) to leverage the currently unused layer2 and layer3 fused maps for improved multi-scale detection. More extensive architectural explorations could involve changing the ResNet-18 backbones for more powerful models like EfficientNetV2, or replacing the detection head with anchor-free alternatives like DETR or two-stage detectors like Faster R-CNN, though this would require significantly more training time and computational power. Finally, for real-world deployment where efficiency is key, the finalized model could be optimized through techniques such as model pruning, quantization, and particularly Knowledge Distillation, which would allow a smaller "student" model to learn from this larger, more complex one, making it suitable for edge devices.

Acknowledgements

This research received support during the Intelligent Monitoring and Control Systems course, instructed by Professor Pasquale Coscia, at the Department of Artifical Intelligence for Science and Technology, University of Milan.

References

Sun, Y., Meng, Y., Wang, Q., Tang, M., Shen, T., & Wang, Q. (n.d.). Visible and infrared image fusion for object detection: A survey.

Ye, Y., Ma, H., Tashi, N., Liu, X., Yuan, Y., & Zihang, S. (n.d.). Object detection based on fusion of visible and infrared images.

Jiang, C., Ren, H., Yang, H., Huo, H., Zhu, P., Yao, Z., Li, J., Sun, M., & Yang, S. (n.d.). M²FNet: Multi-modal fusion network for object detection from visible and thermal infrared images.

Hou, Z., Li, X., Yang, C., Ma, S., Yu, W., & Wang, Y. (n.d.). Dual-branch network object detection algorithm based on dual-modality fusion of visible and infrared images.

Yang, D., Xu, T., Zhang, Y., An, D., Wang, Q., Pan, Z., Liu, G., & Yue, Y. (n.d.). Image-fusion-based object detection using a time-of-flight camera.

Xu, X., Liu, G., Bavirisetti, D. P., Zhang, X., Sun, B., & Xiao, G. (n.d.). Fast Detection Fusion Network (FDFNet): An end-to-end object detection framework based on heterogeneous image fusion for power facility inspection.