ISSN: 2454-9940

INTERNATIONAL JOURNAL OF APPLIED SCIENCE ENGINEERING AND MANAGEMENT

E-Mail :
editor.ijasem@gmail.com
editor@ijasem.org

www.ijasem.org

INTERNATIONAL JOURNAL OF APPLIED SCIENCE ENGINEERING AND MANAGEMENT

ISSN:2454-9940 www.ijsem.org

Vol 20, Issuse.2 April 2026

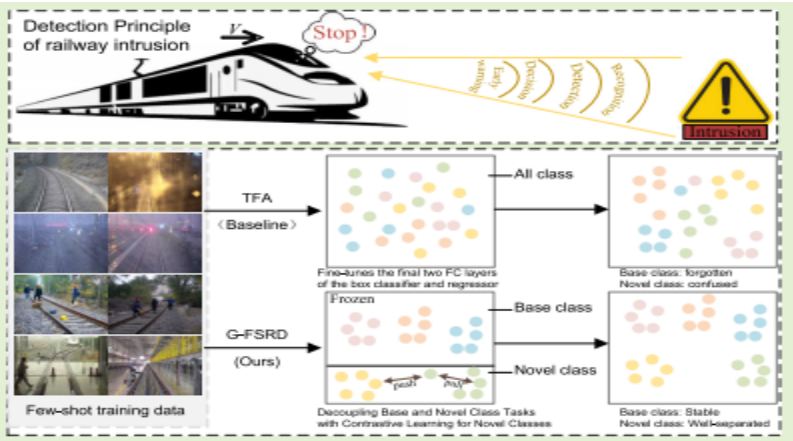
Vision-Based Few-Shot Railway Intrusion Detection via Dual-Detector and Contrastive Learning

[1] L. Hari Krishna Chary,[2] Volkaji Vijay Kumar,[3] Mane Prem,[4] Pothunuri Rajkumar,[5] Nannapuraju Akhila,

[1,] Assistant Professor, Department of ECE, Narsimha Reddy Engineering Collage, Maisammaguda(V), Kompally, Telangana.

[2,3,4,5] Student, Department of ECE, Narsimha Reddy Engineering Collage, Maisammaguda(V), Kompally, Telangana.

Abstract— There has been a meteoric rise in the use of rail transportation, making rail way incursion detection an essential tool for keeping trains running safely. Deep learning and broad object identification frameworks are the current mainstays of railway intrusion detection technologies. Unfortunately, in data-scarce railway contexts, they perform poorly and need costly, large-scale annotated datasets, which drives up data gathering costs. On the other hand, few-shot object detection (FSOD) approaches fail miserably when applied to new classes and experience disastrous losing track of their basis. We provide a few-shot approach for railway intrusion detection using a dual-detector, contrastive learning in new classes (CLNC), and an efficient fine-tuning framework to tackle these difficulties. The vision sensor is a visible-light camera. By separating the detection of new classes from the detection of base classes, the dual-detector architecture reduces the likelihood of catastrophic forgetting, and the CLNC module improves generalization to new classes by increasing intraclass compactness and interclass separability. The effective fine-tuning framework further improves detection accuracy by optimizing module cooperation. Extensive trials on the visible-light camera-collected self-constructed few-shot railway incursion dataset (FSRI2024) show that the proposed G-FSRD outperforms state-of-the-art (SOTA) FSOD approaches. Specifically designed for railway intrusion detection, it maintains common base intrusion detection performance while quickly adjusting to uncommon unique incursions. Visual sensor, few-shot object detection (FSOD), contrastive learning, category forgetting, and railway incursion detection are all terms that may be found in the index.



I. INTRODUCTION

Train safety is becoming more and more important as railway systems develop and operating situations become more complicated [1]. Because of its abruptness, high randomness, and unpredictability, railway incursion has become one of the primary causes endangering the safe functioning of railroads among the many dangers. As a result, several academics and organizations throughout the globe are

now investigating effective techniques of railway intrusion detection [2, 3, 4, 5, 6]. Track switching, level crossings, and areas around station entrances and exits are common locations for rail transport incidents. Trains often travel at speeds lower than 45 km/h in these conditions. Manual monitoring is at the heart of traditional inspection systems, but it may be time-consuming, error-prone, and expensive to implement. Modern railroad safety systems have increasingly high standards for efficiency and dependability, and these constraints make meeting those standards challenging [4]. Here, a flexible and cost-effective method that greatly improves the accuracy and intelligence of railway intrusion detection systems is to integrate visible-light vision sensors with deep learning-based object recognition. While generic object identification approaches based on deep learning have improved upon older methods in terms of flexibility, their performance is still limited by the amount of training data that is available. It is very challenging to gather large-scale, high-quality annotated information because railway infiltration occurrences are so unexpected and unpredictable. Also, there are a limited number of samples for certain kinds of intrusive items, and these things frequently show a long-tail distribution in railway intrusion objects. This means that steady detection performance is a challenge for broad object identification algorithms when data is sparse. Inspired by the speed with which humans can acquire new ideas from very little instances, researchers have developed few-shot object detection (FSOD) algorithms to tackle this difficulty [7, [8], [9], [10], [11]. Even when working with small training sets, these methods strive to keep detection accuracy and generalizability high. The current limitations of generic object detection techniques may be overcome with the help of these approaches, which provide a realistic and effective solution for railway intrusion detection.

There has been little to no widespread application of FSOD approaches to railway intrusion detection as of yet, with much of the research concentrating on public datasets. Also, these approaches have a built-in paradox: they make new classes more generalizable, yet they often cause base classes' detection performance to drastically decline, which leads to catastrophic forgetting. When it comes to the job of railway intrusion detection, this is an intolerable situation. In real life, there are two types of intrusions: frequent base intrusions and uncommon new intrusions. There is a serious threat to user safety if the model loses track of information about base classes, which would significantly reduce the detector's ability to detect typical base incursions. Consequently, there is an immediate need for technologies that may increase the detection performance of frequent base intrusions while simultaneously increasing the detection capabilities of uncommon unique intrusions in railway intrusion detection systems. In order to tackle this difficulty, we thoroughly examine the technique known as the state-of-the-art two-stage fine-tuning approach (TFA) [12]. The few-shot fine-tuning step involves training on a balanced dataset that includes both base and new classes. Each class is given just K-shot samples. The feature distribution of base classes may change since this small dataset does not reflect the real statistical distribution of categories. The outcome might be a change to the decision bounds, which would lead to worse detection performance on base classes. In addition, the detector's decision bounds lean toward base classes because of the unequal distribution of classes, which increases the likelihood of misclassification of fresh class instances as other classes that are visually similar. This further reduces overall detection performance and restricts the model's capacity to generalize to new classes. This paper presents G-FSRD, a few-shot detection framework for railway intrusion detection that is both efficient and brief, to solve the concerns discussed before. An effective fine-tuning framework, a dual-detector, and contrastive learning within new classes (CLNCs) make up G-FSRD. The dual-detector design models the base and new classes individually, which is a unique way to decouple their detection.

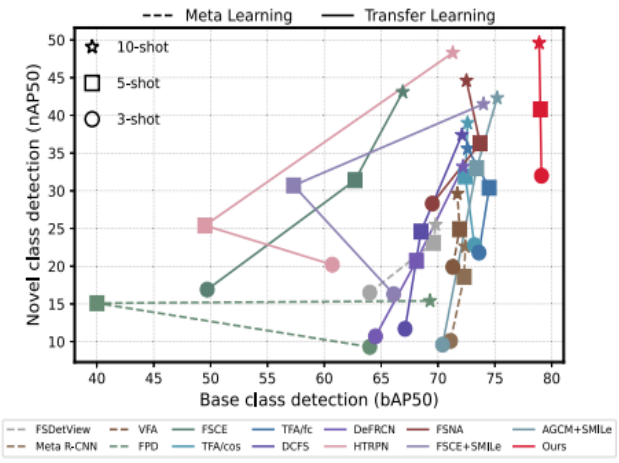


Fig. 1. Performance on FSOD in FSRI2024 under the ten-shot setting, illustrating that our method effectively mitigates catastrophic forgetting in base classes and enhances generalization for novel classes compared to previous methods.

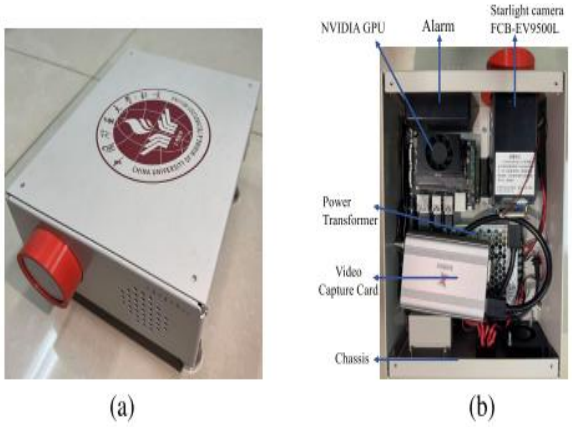


Fig. 2. Prototype for railway intrusion detection. (a) Prototype appearance. (b) Internal structure.

elements that increase the model's flexibility to new classes while keeping the information about base classes, thereby reducing model bias. The novel detector also has a contrastive learning branch to boost the generalizability of new classes; this allows for instance-level feature modeling, which in turn improves intraclass consistency and interclass separability. Lastly, a framework for efficient fine-tuning is developed to maximize the synergy between the CLNCs and the dual-detector. Figure 1 shows experimental findings showing that the suggested strategy greatly enhances the accuracy of detecting uncommon new intrusions while keeping the detection performance of frequent base intrusions. Figure 2 also shows the results of developing a small prototype system for railway intrusion detection. Starlight Camera FCB-EV9500L, video capture card, NVIDIA GPU module, alarm module, and power supply unit (transformer) make up the system. The main image acquisition component, the FCB-EV9500L, takes pictures of the railway environment in real time. The NVIDIA GPU does the intrusion detection after receiving the video feed from the capture card. To improve railway operating safety, the built-in audio-visual alarm module sends out notifications to help with prompt reaction. The video capture card in the prototype transfers live video feeds from the Starlight Camera, which are continuously recording the train environment, to the NVIDIA GPU. The arriving frames are processed by the onboard detection model to identify possible intrusions. The audio-visual alarm module is instantly activated to warn railway staff upon detection. In real-world field deployments, this integrated architecture allows for end-to-end visual monitoring, sophisticated intrusion detection, and timely warning. What follows is an overview of what we contributed to this project. To reduce the impact of model bias and facilitate effective generalization to new classes, we provide a dual-detector module that separates detection of base classes from detection of novel classes.

classes without causing catastrophic forgetfulness while maintaining performance on base classes. 2) The CLNC module is introduced to provide better generalization to new classes. It improves feature representations by using instance-level modeling and similarity assessment of RoI encodings. The detection performance for new classes is enhanced as a result of less intraclass variation and better interclass separability. 3) We provide a technique for detecting intrusions in railway systems using a few shots. Our technique outperforms state-of-the-art SOTA algorithms in experiments conducted on the FSRI2024 dataset, with better detection performance for new classes and stable performance for base classes maintained. Now let's go into the rest of this essay. Research on FSOD and broad object detection is reviewed in Section II. In Section III, we present the G-FSRD framework and how it will be put into action. We lay out the experimental design, measures for assessment, and outcomes in Section IV, and then we analyze and evaluate the data. The essay is concluded and future study objectives are outlined in Section V.

II. RELATED WORKS

Recent efforts on railway intrusion detection are mostly reviewed in this part. The merits and disadvantages of generic object detection algorithms for railway intrusion are thoroughly examined in Section II-A. The FSOD techniques developed for railway incursion are discussed in Section II-B.

A. General Object Detection for Railway Intrusion

The field of railway intrusion detection, which includes both vision-based and nonvisual methods, has come a long way in the last few years. High expenses, complicated installation, and difficult maintenance are common problems with traditional nonvisual approaches. However, because to their low cost and simplicity of implementation, vision based technologies have been gaining more and more attention. One example is the employment of support vector machines (SVMs) [13] for intrusion detection using predetermined color characteristics and aspect

ratios; however, SVMs struggle to recognize objects with multiple scales. In addition to frame difference approaches [15], [16], other techniques such color space transformations, background removal, and linked component analysis [14] have been used. Traditional approaches are not very adaptable to complicated railway situations and mainly depend on bespoke features. Innovations in deep learning have opened up new possibilities for detecting intrusions on railways. With its impressive feature extraction capabilities, the region convolutional neural network (R-CNN) has emerged as a crucial technology in intelligent railroads, and it is the most representative design. Several studies have used it to effectively identify intrusions on railways [17], [18], and [19]. Extensive rail transit datasets have been collected and feature mapped in YOLO [22], [23] utilizing several convolutional layers by Tian et al. [20] and He et al. [21]. The contextual information they extract is still somewhat poor, even though these CNN-based approaches are conceptually basic. Additionally, He et al. [26] improved the global modeling capabilities for multiscale railway obstacle features by combining Mask-RCNN [24] with the Swin Transformer (SwinT) [25]. Even with single-stage approaches, remarkable improvement has been made [27], [28], and even more recently with anchor-free methods [29]. One alternative is to train intrusion detection classifiers, such the enhanced VGG Net, utilizing drone-collected aerial video images [30]. Drone detection, on the other hand, is much more expensive than traditional camera surveillance systems. There are a number of object detection algorithms that our team has created that are based on deep learning. In particular, the authors of [2, 4], [17], [31], and [32] aimed to strengthen detection resilience, model compactness, and inference efficiency in the face of complicated railway situations. Nevertheless, these techniques need labor-and time-intensive approaches to acquire large-scale, well labelled datasets. For a number of reasons, samples might be hard to come by in complicated railway applications. FSOD is well-suited for railway intrusion detection as it does not depend on massive datasets.

B. Few-Shot Object Detection

FSOD's primary goal is to make it easier to train object detectors with less samples. Current FSOD research may be categorized into two main methods: meta-learning approaches [33, [34], [35], [36] and fine-tuning-based strategies [12], [37], [38], [39], [40], [41], [42]. Using a staged and iterative meta-learning framework, meta-learning-based techniques train a meta-learner to help transfer information from base courses. There is a direct correlation between the increasing computational complexity caused by methods such as Meta-R-CNN [34] and FSDetView [36], which reweight RoI features for each class using attention vectors. Lightweight meta-learning model FSRW [33] builds on YOLOv2 [43] by using a re-weighting module and a meta-feature learner to extract features that can recognize new classes of objects. A typical issue with the previously described meta-learning approaches is the substantial increase in computing cost caused by the extra support branch. The support and inquiry branches add n times more processing cost in an N-way setup, rendering them unfeasible for detecting different types of railway infiltration. Rail intrusion detection, on the other hand, is best handled by approaches based on fine-tuning, which can simply accomplish full-class detection. In their proposal of the MPSR technique, Wu et al. [39] successfully addressed the problem of scale variation by including multiscale positive samples as input. With the use of an all-attention strategy, FSNA [42] significantly improves performance on new classes by making object identification more resilient via neighborhood information adaptation and by extracting more complete and improved feature representations. To improve FSOD's class separability and generalizability and to speed up convergence across different detection frameworks, SMILe [44] implements submodular mutual information losses. While TFA [12] did help slow down the fine-tuning process and improve generalization for new classes, it only fine-tuned the detector's last layer during the second step of fine-tuning, which meant that the performance of base classes was still degraded. Research on the use of FOSD for railway intrusion detection is still in its early stages. A state-of-the-art few-shot learning method developed specifically for railway video intrusion detection was presented by Gong et al. [3]. However, this approach can only be used for intrusion classification and cannot be used for geographical localization. Nevertheless, research shows that current FSOD approaches always have terrible base class forgetting, even while they are good at recognizing new classes after few-shot training. These models significantly degrade performance when used to identify typical base-class intrusions in railway intrusion detection situations. This technological shortcoming might compromise the safety protection systems of railway infrastructure and directly impact railway operating safety. Consequently, we investigate and enhance TFA by implementing generalized FSOD, with the goal of achieving high generalization for uncommon new

intrusions and strong retention of common base intrusions in railway intrusion detection. I. PROCEDURE In Section III-A, we lay up the groundwork for FSOD, then in Section III-B, we go back to the TFA approach. To address the issue of model bias, a dual-detector is suggested in Section III-C. Section III-D presents CLNCs as a means to enhance the capacity to generalize to new classes. Section III-E presents an efficient framework for fine-tuning in detail. The design and implementation of G-FSRD's loss function are described in Section III-F.

A. Preliminaries

The goal of FSOD is to train detectors to identify new classes with less training data, so they can quickly adapt to different detection tasks. According to earlier research [12], [33], it is possible to divide the dataset into two parts: the base dataset, Cbase, and the new dataset, Cnovel. Usually, Cbase has many annotated examples of frequent base intrusions, whereas Cnovel only has K-shot samples of uncommon novel intrusions (where K is less than or equal to 10). Crucially, there is no overlap between Cbase and Cnovel (Cbase ∩ Cnovel = ∅). Figure 3 shows the two main phases of the training process: the basic phase and the fine-tuning phase. A large number of common base intrusion samples are used for the base training stage, whereas a balanced dataset Dfew is used for the fine-tuning step. This dataset contains both frequent and rare intrusions, but each class only has K-shot examples. In few-shot classification, the traditional N-way K-shot strategy is commonly N = 1, 5. However, in our evaluation of the model, we use a more difficult full-way (N = ‖CbaseUCnovel‖) K-shot scheme. Our goal is to make the detector better at detecting uncommon new intrusions with smaller samples without affecting its ability to identify frequent base intrusions.

B. Revisiting TFA

A commonly used baseline approach for FSOD that relies on fine-tuning is TFA [12]. This model employs a two-stage training framework: first, the base training, where the model is trained on a dataset with abundant annotations for base classes; second, the fine tuning, where the model is trained on a balanced dataset with K annotated samples per class, comprising both base and novel classes. To help the model identify new categories, the classifier weights associated with them are initially set to random during the fine-tuning step. After that, the balanced dataset is used to fine-tune the RoI head's last layer.

A cosine similarity-based classifier is also used in the second step of fine-tuning by TFA [12]. The cosine similarities between the input features F(x) and the weight vectors of each category are represented by the scaled similarity score matrix S, which is produced by the classifier. This is seen in the following equation:

$$s_{i,j} = \frac{\alpha \mathcal{F}(x)_i^{\top} w_j}{\|\mathcal{F}(x)_i\| \, \|w_j\|} \quad (1)$$

where si, j is the score that indicates how similar the ith object proposal from the input x is to the class prototype wj ∈ R d. We have set the scaling factor α at 20. The cosine similarity-based classifier improves detection performance on new classes while retaining accuracy on base classes by performing instance-level feature normalization, which substantially minimizes intraclass variation. The TFA [12] technique is well-suited to our goal of allowing coexistence identification for both base and unique categories due to its inherent simplicity and high overall performance on the FSRI2024 dataset. So, we

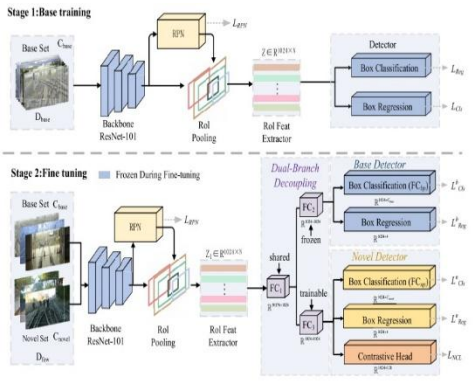


Fig. 3. Overall architecture of the proposed G-FSRD. In Stage 1, a standard Faster R-CNN is trained on base classes to produce a base detector. In Stage 2, a dual-detector is constructed by duplicating and decoupling the RoI head into two parallel branches: a frozen base detector and a trainable novel-detector. The novel detector includes an additional contrastive head, which performs supervised contrastive learning solely within novel classes RoIs.

start with TFA [12] and suggest G-FSRD, a better method that also uses a two-stage tuning paradigm. For FSOD in railway incursion situations, G-FSRD provides an effective fine-tuning framework.

C. Dual-Detector

As mentioned in Section I, the catastrophic forgetting issue causes a significant drop in base class performance during the fine-tuning step in the majority of existing FSOD approaches. This is because the balanced dataset Dfew used for fine-tuning does not include K-shot Cbase instances.

show how Cbase is really distributed from the training set, which has extensive annotations. Eventually, just the K-shot generalization remains after training has progressed, as the impact of Cbase employed in base training decreases. In addition, predictions for base classes could be impacted by the random initialization of weights for novel classes. Overestimation of new classes by the Softmax classifier, caused by insufficient training data, results in lower confidence ratings for base classes. We developed a dual-detector module (Fig. 3) to solve this problem. We model the properties of both the base and novel classes separately, in contrast to TFA [12], which combines the two jobs into one. To improve upon the original base detector, we build an extra detector to identify new classes. When training new courses, the base detector remembers all the important information, and when training novel classes, the novel detector learns and fine-tunes the information. In particular, following RoI pooling, there are three layers: FC1, FC2, and FCp, which is the prediction layer. A new fully-connected layer, FC3, is created during the fine-tuning step and functions in tandem with FC2, being a copy of the FC2 layer. All three of FC2 and FC3 share FC1's weights. Together, FC2 and its matching prediction layer FCp (formerly called FCbp) are responsible for making predictions based on the base classes. Both layers remember the weights learnt in the base training stage, thus the information from the base classes is kept. Predictions for new classes are made using FC3 and its corresponding prediction layer (FCnp), with the weights of these layers changed during tuning to allow for successful generalization to new classes. Separate Softmax activations are applied to the FCbp and FCnp outputs to further reduce interference between the base and novel classes. In addition to boosting the generalizability of new classes, this architecture guarantees the stability of detecting base classes.

D. Contrastive Learning Within Novel Classes

Inadequate generalization of new classes occurs when the detector incorrectly identifies occurrences of novel classes as perplexing classes during fine-tuning. In few-shot railway infiltration detection, this problem stands out.
Our proposal is a CLNC module that will solve this problem. The clipped proposal features and their ground-truth objectness labels are inputted into an encoder branch, as shown in Figure 4, drawing inspiration from references [40] and [45]. Within the proposal embedding space, this branch precisely records the degree of similarity and dissimilarity across classes at the instance level. More specifically, the innovative detector's RoI head incorporates a constructive head that runs in tandem with the regression and classification branches. Encoding RoI characteristics into 128-D embeddings, this contrastive head is composed of a multi-layer perceptron (MLP). In order to maximize a supervised contrastive loss, which promotes better separation between classes and closer clustering of same-class suggestions, we compute the pairwise similarities among these embeddings. As opposed to worldwide

While other contrastive strategies, like FSCE [40], build positive and negative pairs from all regions of interest (RoIs), including those from base classes and background regions, our CLNL module limits contrastive learning to RoIs of novel classes, excluding base-class and background proposals explicitly. The optimization noise and semantic

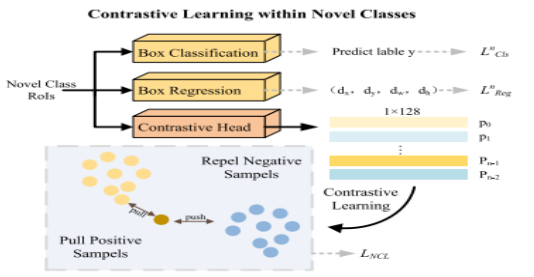


Fig. 4. To incorporate a contrastive branch within the novel detector, encode the RoI features into 128-D feature vectors for contrastive learning. This approach aims to pull features of the same class closer together while pushing features of different classes apart.

interference prevalent in global contrastive contexts are mitigated by this method. In our CLNL module, pairs of new class labels are used to build positive pairings, while pairs of non-novel class labels are used to construct negative pairs, all inside the same mini-batch. In order to maintain semantic consistency and improve optimization stability, RoIs from base classes or backgrounds are purposefully ignored. Furthermore, as seen in (2), we developed a contrastive loss function LNCL specifically for the contrastive branch. To be more precise, P is the number of RoI features that are input into the RoI head, and {ci, yi} i P = 1 is the set of RoI features that consist of ground truth labels and contrastive encodings. Here, ci is the feature that is generated by the ith region proposal, and yi is the ground truth label that is associated with it. Q is the number of suggestions with the same label as yi. This parameter, usually set to $\tau = 0.2$, acts as a temperature coefficient and is a hyperparameter [46]. Its main role is to

control how well the model can distinguish between negative samples.

$$\mathcal{L}_{\mathrm{NCL}} = -\frac{1}{P}\frac{1}{(Q-1)}\sum_{i=1}^{Q} C_i$$

$$D_i = \sum_{k=1}^{P}\prod_{(k\neq i)} \exp\left(\tilde{c}_i \cdot \tilde{c}_j/\tau\right)$$

$$C_i = \sum_{j=1, j\neq i}^{Q}\prod_{(y_i=y_i)} \log\left[\frac{\exp\left(\tilde{c}_i \cdot \tilde{c}_j/\tau\right)}{D_i}\right]$$

c˜ is a measure of the cosine similarity between the ith and jth proposals, and it is defined as c/‖c‖, representing feature normalization. The indicator function Q (yi = y j) gives 1 when yi = y j and 0 otherwise. It also represents an additional variable. The aforementioned loss function improves inter-class distinction while simultaneously increasing intraclass similarity within a given class. As a result, examples of the same class are compressed into a smaller group, while instances of other classes are compressed into larger gaps.

E. Efficient Fine-Tuning Framework

Figure 3 shows the two main phases of our G-FSRD training framework: foundation training and fine-tuning. The first method requires training on a massive dataset with lots of annotations (called Cbase), while the second method involves fine-tuning on a balanced dataset with both Cbase and Cnovel but with less annotations so that the previous knowledge from base training can be transferred more effectively. Continuing with earlier SOTA approaches [12], [34], [36], [37], [38], [40], [41], [48], we use Faster R-CNN [47] as our baseline detector. To find the best fine-tuning framework for our G-FSRD, we studied the Faster R-CNN components in detail and ran TFA-based ablation experiments (see Section IV-C) [12] (inspiring by previous fine-tuning based methods [12], [37], [38], [40], [41]). To start, we don't tweak the backbone's settings since doing so would create catastrophic amnesia and the characteristics it extracts are shallow. Secondly, we optimize the RPN, a binary classifier that does not care about classes, to provide candidate areas. To improve the quality of the proposals, we use some new samples to tweak their decision bounds and increase the NMS threshold so that they keep more positive anchors. The prediction results are greatly influenced by the RoI head. Compared to the typical RoI batch size of 512, we reduced the number of suggestions used for loss calculation in the RoI head to 128. Although performance on Cbase may suffer, performance on new classes is vastly improved when the RoI head and RPN are fine-tuned together. We still benefit from combined RPN and RoI head fine-tuning since the dual-detector can keep the Cbase's performance. Algorithm 1 lays out the suggested fine-tuning framework in full for your perusal. The first step is to minimize the combined loss of LRPN, LReg, and LCls while training the base detector fbase on the base classes Cbase. This process continues until convergence. After that, the weights of the converged fbase are used to establish the joint detector fall. It is then fine-tuned using the balanced dataset of Cbase and Cnovel, where each class comprises only K-shot examples. An important step in avoiding catastrophic forgetting is to freeze the detection loss of the base classes LDet B at this time. Reduced normalized contrastive loss (LNCL), reduced loss (L n Cls), and reduced loss (L n Reg) optimize the loss of the new classes (LDet N). The last step in updating the joint detector fall is to minimize the total loss, which includes L RPN, LDet B, and LDet N. This process ensures that the detection capacity is balanced for both base and novel classes. In Section III-F, we

Algorithm 1 Efficient Fine-Tuning Framework

Require:
N, minibatch size
f_{base}, f_{novel}, f_{all}, detectors
α, balance factor, the weight of $\mathcal{L}_{\mathrm{NCL}}$
lr_{base}, lr_{novel}, lr_{all}, learning rates

// Train f_{base} on C_{base}, where each class contains sufficient samples
repeat
  Iteratively sample minibatch $\mathcal{X}_{\mathrm{base}} = \{x_i\}_{i=1}^{N}$
  Compute $\mathcal{L}_{\mathrm{base}} \leftarrow \mathcal{L}_{\mathrm{RPN}} + \mathcal{L}_{\mathrm{Reg}} + \mathcal{L}_{\mathrm{Cls}}$
  Update $f_{\mathrm{base}} \leftarrow f_{\mathrm{base}} - lr_{\mathrm{base}} \cdot \nabla_{f_{\mathrm{base}}} \mathcal{L}_{\mathrm{base}}$
until f_{base} converges
// Train f_{all} on $C_{\mathrm{base}} \cup C_{\mathrm{novel}}$, where each class contains only K-shot samples
Initialize f_{all} with the weights of the converged f_{base}
repeat
  Iteratively Sample minibatch $\mathcal{X}_{\mathrm{all}} = \{x_i\}_{i=1}^{N}$
  if x_j belongs to Base classes **then**
    Compute $\mathcal{L}_{\mathrm{Det}}^{B} \leftarrow \mathcal{L}_{\mathrm{base}}$
    Copy the parameters: $f_{\mathrm{base}} \leftarrow f_{\mathrm{base}}$
  else
    Compute $\mathcal{L}_{\mathrm{Det}}^{N} \leftarrow \mathcal{L}_{\mathrm{Cls}}^{n} + \mathcal{L}_{\mathrm{Reg}}^{n} + \alpha\mathcal{L}_{\mathrm{NCL}}$
    Update $f_{\mathrm{novel}} \leftarrow f_{\mathrm{novel}} - lr_{\mathrm{novel}} \cdot \nabla_{f_{\mathrm{novel}}} \mathcal{L}_{\mathrm{Det}}^{N}$
  end if
  Compute $\mathcal{L} \leftarrow \mathcal{L}_{\mathrm{RPN}} + \mathcal{L}_{\mathrm{Det}}^{B} + \mathcal{L}_{\mathrm{Det}}^{N}$
  Update $f_{\mathrm{all}} \leftarrow f_{\mathrm{all}} - lr_{\mathrm{all}} \cdot \nabla_{f_{\mathrm{all}}} \mathcal{L}$
until f_{all} converges

provide the particular loss functions.

F. Training Loss

A combination of the base detector loss function (LDet B), the binary cross-entropy loss (LRPN), and the total loss (L) is the result.

novel detector loss function (*L*Det *N*). It is detailed as follows:

$$\mathcal{L} = \mathcal{L}_{\text{RPN}} + \mathcal{L}^{B}_{\text{Det}} + \mathcal{L}^{N}_{\text{Det}} \tag{5}$$

$$\mathcal{L}_{\text{RPN}} = \mathcal{L}_{\text{obj}} + \mathcal{L}_{\text{bbox}} \tag{6}$$

LRPN stands for the RPN's loss function, Lobj for object classification's binary cross-entropy loss, and Lbbox for bounding box regression's smooth-L1 loss. The loss function for binary cross-entropy is

$$L = -\frac{1}{N}\sum_{i=1}^{N}\left[y_i \log\left(p_i\right) + \left(1 - y_i\right) \log\left(1 - p_i\right)\right]$$

in where N is the total number of samples, yi is the class of the ith sample, and pi is the projected value of the ith sample. Here is the formula for the smooth-L1 loss function:

$$\text{smooth}_{L_1}(x) = \begin{cases} 0.5x^2, & \text{if } |x| < 1 \\ |x| - 0.5, & \text{otherwise} \end{cases}$$

in which x is the numerical disparity between the ground-truth box and the anticipated box. This is how the LDet B and LDet N loss functions are articulated:

$$\mathcal{L}^{B}_{\text{Det}} = \mathcal{L}^{b}_{\text{Cls}} + \mathcal{L}^{b}_{\text{Reg}} \tag{9}$$

$$\mathcal{L}^{N}_{\text{Det}} = \mathcal{L}^{n}_{\text{Cls}} + \mathcal{L}^{n}_{\text{Reg}} + \alpha\,\mathcal{L}_{\text{NCL}} \tag{10}$$

the contrastive loss function, LNCL, is defined as follows: L b Reg and L n Reg are smooth-L1 loss functions for regression, L b Cls and L n Cls are cross-entropy losses used for bounding box classification, and

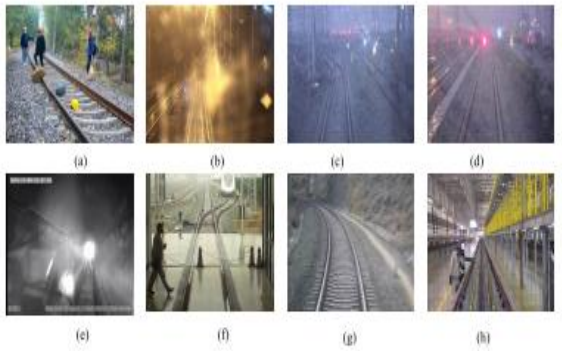


Fig. 5. Representative samples from the FSRI2024 dataset, covering diverse weather conditions (e.g., day, night, rain, and fog) and functional railway scenes (e.g., tunnel, station shunting, field driving, and multiobjective environments). (a) Day. (b) Night. (c) Rainy weather. (d) Foggy weather. (e) Tunnel. (f) Station shunting. (g) Field driving. (h) Multiobjective obstacles.

throughout the novel's course of study. In addition, we discovered that the training stability may be maintained when the novel detector loss is directly added to the novel classes' contradictory loss in a multitask way. In order to compensate for the contrastive learning loss, we put the weighting factor at α = 0.6. One way to think about the loss function is as

$$H\left(p, q\right) = -\sum_{i=1}^{n} p\left(x_i\right) \log\left(q\left(x_i\right)\right) \tag{11}$$

the model's confidence in predicting xi is measured by log(q(xi)), where n is the total number of classes, p(xi) is the real probability distribution of sample xi, and q(xi) is its forecast probability.

IV. EXPERIMENTS

We begin by outlining the experimental setup and the built FSRI2024 dataset in Section IV-A. Next, Section IV-B showcases the experimental findings on FSRI2024, proving that our strategy is successful. Ablation studies and visualization analyses are provided in Section IV-C for additional analysis of our technique. Section IV-D concludes by testing the suggested model's generalizability using the Pascal VOC dataset.

A. Benchmarks and Setups

1) Dataset: We built FSRI2024, a few-shot rail way incursion dataset, by following the classification and Split settings of the Pascal VOC dataset, which has been widely used in previous FSOD efforts [12], [33]. The data was gathered by means of a prototype

system that included a Starlight FCB-EV9500L visible-light camera (1920 × 1080, 30 FPS). This system could be installed inside train cabins or left at certain spots along the track. The dataset includes a variety of weather conditions (sunny, wet, and foggy), time periods (morning to night), and spatial settings (e.g., platforms, tunnels, and natural paths) to guarantee environmental diversity (Fig. 5). Nine safety-critical categories are included in FSRI2024: Rail way Left, Railway Right, Railway Straight, Bullet Train, Pedes train, Helmet, Tool Kit, Spanner, and Rockfall. The remaining categories aid in collision prevention, while track-type recognition (RailwayLeft/RailwayRight/Straight) provide directional signals.

TABLE I
OVERVIEW OF PER SPLIT FOR OUR EXPERIMENTS

Split	Base Classes (C_{base})	Novel Classes (C_{novel})
1	Pedestrian, BulletTrain, RailwayLeft, RailwayRight, Spanner, ToolKit	Helmet, RailwayStraight, Rockfall
2	Pedestrian, RailwayStraight, RailwayRight, Helmet, Spanner, Rockfall	ToolKit, BulletTrain, RailwayLeft
3	BulletTrain, RailwayStraight, RailwayLeft, Helmet, Spanner, ToolKit	Rockfall, Pedestrian, RailwayRight

As shown in Table I, we apply VOC-style three-way class splitting to improve assessment robustness and decrease bias induced by particular class divisions. A single procedure was followed by a trained staff as they manually executed all annotations using the LabelMe application. Expert auditing and peer review made sure that the annotators were consistent with each other. Figure 6 shows that FSRI2024 follows a traditional long-tailed class distribution, which is in line with what would happen in a real-world incursion. Random cropping, horizontal flipping, and brightness/contrast manipulation are examples of traditional TFA-style data augmentation [12] that we used during training to promote generalization while being consistent with earlier FSOD methods. Lastly, we tested the FSOD approaches on a dataset of 27,85 pictures, which included 14,381 incidents. Figure 7 displays the test set's instance counts and average area ratios (AARs) for each class. Pedestrians and Helmets have the highest counts, with over 3,000 instances apiece; all other classes have counts over 400. Helmet, ToolKit, and Spanner all have AAR values below 2%, which is considered tiny item, thus proving that the suggested approach is good for detecting small objects. With more test examples, a wider scope, and more complicated sceneries, FSRI2024 presents a greater challenge than VOC. Section 2: Assessment Procedures We are mostly concerned with how well Cnovel and Cbase function as a whole. Our goal in reporting the mean average precision (mAP50) for all classes at an intersection over union (IoU) criterion of 0.5 is to assess the detection performance on FSRI2024. Furthermore, we document the nAP50 and bAP50 for every Cnovel and Cbase, correspondingly. Here are the definitions of precision and recall: true positives (TP), false positives (FP), and false negatives (FN) are the three possible outcomes. By using 11-point interpolation, the average precision (AP) may be calculated from the precision-recall (P-R) curve. The mean AP50 is determined by averaging the AP across all classes at an IoU threshold of 0.5.

$$\mathrm{mAP}_{50} = \frac{1}{|C|} \sum_{c \in C} \mathrm{AP}_{50}(c). \tag{14}$$

To separately evaluate base and novel categories, we define

$$\mathrm{bAP}_{50} = \frac{1}{|C_{\mathrm{base}}|} \sum_{c \in C_{\mathrm{base}}} \mathrm{AP}_{50}(c) \tag{15}$$

$$\mathrm{nAP}_{50} = \frac{1}{|C_{\mathrm{novel}}|} \sum_{c \in C_{\mathrm{novel}}} \mathrm{AP}_{50}(c). \tag{16}$$

To ensure a level playing field, we ran comprehensive comparison studies on one, two, three, five, and ten shots on three separate FSRI2024 splits. In addition, an extensive set of tests was run on Pascal VOC to assess G-FSRD's generalizability. Section 3: Details of the Implementation: The suggested approach employs a ResNet-101 [49] backbone pretrained on ImageNet, with Faster R-CNN [47] serving as the core detection framework. Our G-FSRD first randomly initializes and then fine-tunes all of the layers used for fine-tuning. In order to optimize our network beginning to finish, we use stochastic gradient descent (SGD) with the following parameters: a batch size of 16, a momentum parameter of 0.9, and a weight decay of 1e−4. During the base training stage, the learning rate is initially set to 0.02, and then fine-tuned to 0.01. The ten-, five-, and three-shot iterations of training are 6,000, 4,000, and 3,000 iterations, respectively, during fine-tuning.

INTERNATIONAL JOURNAL OF APPLIED SCIENCE ENGINEERING AND MANAGEMENT

ISSN:2454-9940 www.ijsem.org

Vol 20, Issuse.2 April 2026

After a certain amount of rounds, the learning rate is reduced by 0.1. The computational configuration used for all studies consisted of four 11 GB NVIDIA GeForce GTX 1080Ti GPUs.

B. Results on FSRI2024

We performed quantitative and qualitative studies using the FSRI2024 dataset to assess G-FSRD's performance.

Firstly, we compared our technique with methods based on fine-tuning [12], [37], [38], [40], [41], [42], and [44], as well as methods based on meta-learning [34], [35], [36], and [48]. All of the FSRI2024 classes' mAP50 findings from the different approaches are shown in Table II. The top two mAP50 results are highlighted in red and blue, respectively, to indicate their excellence. Our technique outperforms current SOTA methods in the majority of scenarios (i.e., K = 1, 2, 3, 5, and 10 shots), as shown. It seems that fine-tuning tactics are both easier and more successful than meta-learning-based approaches, as methods based on fine-tuning produced more best and second-best outcomes. In the same vein as several other strategies for fine-tuning, G-FSRD

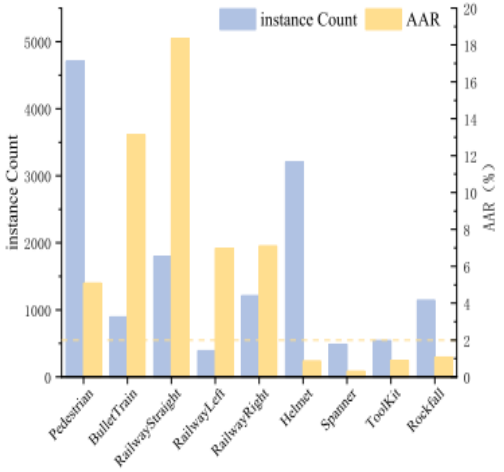


Fig. 7. Number of instances for each class and their AAR values in the test set. The constructed dataset is larger and more complex in terms of quantity and scene complexity compared to the FSOD benchmark Pascal VOC, making it more challenging.

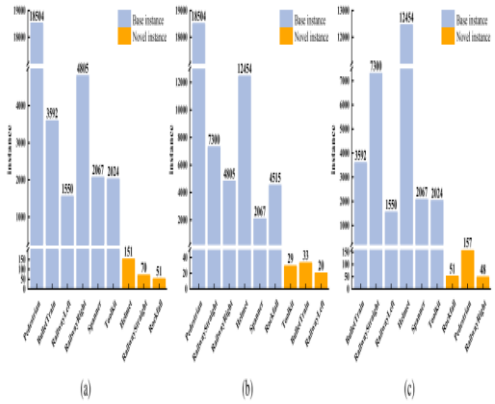


Fig. 6. Long-tailed distribution of object instances in FSRI2024 across different Splits. Base classes contain a large number of instances, while novel classes are limited to ten shots per category. (a) Split 1. (b) Split 2. (c) Split 3.

utilizes TFA [12] as a foundation because to its ease of use and outstanding performance. In particular, on Splits 1 and 2, G-FSRD receives an improvement of 15.2-25.9 points in one-shot and two-shot settings. The accuracy of railway intrusion detection is greatly improved, and the average accuracy is increased by 5.3-10.6 points in different setups. Table III shows that our technique still performs competitively, even if we are not primarily concerned with unique class performance. The generalization process to uncommon new intrusions causes FSCE [40], HTRPN [37], and FSNA [42] to severely forget frequent base intrusions, even if they have shown competitive performance. Unfortunately, this leads to an unacceptable drop in detection performance for typical base invasions. Because they estimate the whole probability distribution of RoI using features re-weighted by particular class attention vectors, meta-learning-based approaches can't generalize to uncommon new intrusions. Although fine-tuning-based approaches perform worse with smaller samples, they get better with larger ones. As a result, our strategy is advantageous for accomplishing incremental detection and identifying uncommon unique incursions.

We also present experimental results for all base classes to demonstrate our method's nonforgetting performance on common base incursions. Table IV displays the mean detection accuracy of each base class across all configurations. Split 3, Split 2, and Split 1 all achieved bAP50 scores of 79.2, 80.5, and 77.9, respectively, at the base stage of G-FSRD. The experimental data show that our strategy significantly reduces catastrophic forgetting. G-FSRD is resilient

to changes in sample size as it keeps bAP50 performance constant across all splits. This means that the few-shot detector keeps all of the useful information from the base classes dataset, no matter how small the sample is. The effectiveness of FSCE [40], HTRPN [37], and FSNA [42] saw a significant drop when applied to base classes, despite their promising performance on new classes. Notably, HTRPN [37] on Split 2 is not appropriate for our objectives since their bAP50 averages 46.4 points lower than G-FSRD. While the highly competitive TFA [12] did a better job of recovering the performance of the base classes as the sample size rose, it was still far behind our technique. In Split 1, for instance, G-FSRD had a bAP50 that was 2.2 points higher than the ten-shot group. Noteworthy, compared to approaches based on fine-tuning, meta-learning methods keep base classes detection performance better. However, meta-learning methods are less affected by sample size, meaning that performance doesn't improve as much as it does with fine-tuning methods. To summarize, the detection performance for typical base invasions may be maintained using our technique. Secondly, we performed a battery of qualitative studies to provide a more intuitive illustration of G-FSRD's advantage. All qualitative trials were conducted using the ten-shot setting of Split 1 unless otherwise noted. Fig. 8 shows the three approaches' precision-recall (PR) curves for nine different types of intrusions: TFA [12], HTRPN [37], and the proposed G-FSRD. In Figure 8, we can see that the

TABLE II

RESULTS OF FSOD METHODS ON THE FSRI2024 DATASET FOR ALL CLASSES $C_{base} \cup C_{novel}$ (MAP50) ACROSS THREE SPLITS. THE BEST AND SECOND-BEST RESULTS ARE MARKED IN **RED**/**BLUE**, RESPECTIVELY. OUR METHOD CONSISTENTLY ACHIEVES BETTER PERFORMANCE THAN THE BASELINE ACROSS DIFFERENT SHOTS

Type	Method/Shots	All Split 1					All Split 2					All Split 3				
		1	2	3	5	10	1	2	3	5	10	1	2	3	5	10
Meta Learning	FSDetView [36]	37.9	45.6	48.3	54.1	55.0	25.1	39.5	42.2	49.6	51.6	30.5	37.8	40.9	47.1	48.6
	Meta R-CNN [34]	46.5	50.7	50.8	54.5	55.8	35.6	44.6	46.1	50.0	53.3	40.2	42.8	44.0	48.5	48.4
	VFA [48]	40.7	50.2	53.5	56.2	57.6	35.3	40.1	40.9	46.4	50.7	**45.8**	**48.9**	48.7	51.7	54.3
	FPD [35]	39.4	40.9	45.8	31.7	51.3	22.8	36.3	40.6	43.6	53.1	36.1	35.9	38.9	41.1	48.6
Fine Tuning	TFA w/cos [12]	45.8	51.8	**56.4**	58.9	61.4	36.7	44.7	47.6	54.6	56.3	35.0	43.8	44.7	50.7	54.2
	TFA w/fc [12]	**47.1**	52.9	56.3	59.8	63.0	35.4	47.0	**51.9**	54.6	56.0	31.6	39.5	49.1	49.8	55.0
	FSCE [40]	43.6	48.2	38.8	52.3	59.0	**37.6**	**49.2**	25.4	29.7	50.1	36.1	44.8	30.7	40.0	52.6
	DCFS [41]	42.4	48.1	48.6	53.9	60.5	28.9	44.8	44.3	50.4	56.2	29.7	42.9	45.7	48.9	55.0
	DeFRCN [38]	43.0	47.1	46.6	52.3	59.2	32.4	41.1	44.6	51.9	57.1	28.7	44.1	45.9	50.9	55.5
	HTRPN [37]	30.2	42.3	47.2	41.4	61.1	18.2	26.5	29.5	37.1	43.8	27.0	34.4	44.0	36.0	58.8
	FSNA [42]	46.4	**53.8**	55.8	**61.2**	63.2	23.7	44.9	48.1	53.0	56.8	40.0	47.9	50.9	53.0	57.9
	FSCE+SMILe [44]	42.4	47.5	49.5	57.3	63.2	29.2	42.3	43.4	54.5	59.0	37.8	43.6	45.8	53.2	59.2
	AGCM+SMILe [44]	45.7	50.2	54.2	59.9	**64.3**	37.5	46.5	47.4	**58.0**	**60.8**	41.3	48.9	**51.5**	**55.9**	**59.4**
	G-FSRD(Ours)	**54.3**	**58.2**	**63.4**	**66.3**	**69.1**	**61.3**	**62.2**	**62.5**	**64.6**	**65.1**	**55.7**	**56.4**	**58.6**	**60.2**	**63.0**

TABLE III

RESULTS OF FSOD METHODS ON THE FSRI2024 DATASET FOR NOVEL CLASSES C_{novel} (NAP50) ACROSS THREE SPLITS. THE BEST AND SECOND-BEST RESULTS ARE MARKED IN **RED**/**BLUE**, RESPECTIVELY. WHILE NAP50 IS NOT THE PRIMARY FOCUS OF OUR STUDY, OUR METHOD DEMONSTRATES COMPETITIVE PERFORMANCE

Type	Method/Shots	Novel Split 1					Novel Split 2					Novel Split 3				
		1	2	3	5	10	1	2	3	5	10	1	2	3	5	10
Meta Learning	FSDetView [36]	0.5	9.9	16.5	23.1	25.5	9.1	12.1	13.6	13.3	14.8	6.1	5.5	6.2	9.2	10.9
	Meta R-CNN [34]	0.5	9.9	10.1	18.6	22.6	10.4	10.5	12.0	13.9	18.2	8.7	6.7	3.8	7.3	11.9
	VFA [48]	2.2	11.5	19.9	24.9	29.6	6.5	10.4	10.2	8.7	11.9	6.6	10.0	10.5	10.2	19.4
	FPD [35]	0.5	6.8	9.3	15.1	15.4	12.0	10.9	15.1	20.6	25.7	3.0	9.0	10.3	9.2	19.1
Fine Tuning	TFA w/cos [12]	0.6	11.8	22.8	31.9	39.0	16.8	**17.4**	21.0	28.2	30.9	0.2	2.1	5.9	9.7	14.6
	TFA w/fc [12]	0.4	11.2	21.8	30.4	35.6	**17.0**	16.4	**21.5**	28.4	30.3	0.1	3.4	3.1	8.2	15.8
	FSCE [40]	1.3	14.4	16.9	31.4	43.1	16.6	16.6	19.6	**32.5**	**38.4**	8.4	7.3	13.1	17.7	31.3
	DCFS [41]	1.6	9.1	11.7	24.6	37.4	14.1	15.3	15.8	20.5	27.6	9.1	9.5	7.7	10.5	23.0
	DeFRCN [38]	1.1	9.1	10.7	20.7	33.2	15.4	16.7	17.9	24.8	29.7	6.1	9.3	11.5	13.3	24.4
	HTRPN [37]	3.1	14.9	20.2	25.4	**48.3**	15.6	17.1	19.7	31.9	**40.8**	6.6	12.1	**18.5**	20.7	**38.1**
	FSNA [42]	**5.1**	**17.2**	**28.3**	**36.3**	44.6	14.5	11.9	18.3	26.8	29.1	**10.3**	**12.9**	**13.6**	15.7	24.1
	FSCE+SMILe [44]	**5.1**	8.1	16.3	30.7	41.5	15.3	15.7	17.3	29.0	37.3	8.6	3.9	15.7	**21.6**	30.0
	AGCM+SMILe [44]	**4.5**	10.2	9.6	33.0	42.3	15.6	17.1	19.3	30.6	34.0	7.7	**14.4**	17.7	21.1	24.9
	G-FSRD(Ours)	**5.1**	**16.9**	**32.0**	**40.8**	**49.6**	**22.7**	**25.6**	**26.7**	**32.8**	34.5	**9.5**	11.3	**18.5**	**24.1**	**33.0**

INTERNATIONAL JOURNAL OF APPLIED SCIENCE ENGINEERING AND MANAGEMENT

ISSN:2454-9940 www.ijsem.org

Vol 20, Issuse.2 April 2026

TABLE IV

RESULTS OF FSOD METHODS ON THE FSRI2024 DATASET FOR BASE CLASSES C_{base} (BAP50) ACROSS THREE SPLITS. THE BEST AND SECOND-BEST RESULTS ARE MARKED IN **RED**/**BLUE**. OUR METHOD DEMONSTRATES GOOD BASE CLASS RETENTION PERFORMANCE AND SHOWS FURTHER IMPROVEMENTS COMPARED TO OTHER BASELINES

Type	Method/Shots	Base Split 1					Base Split 2					Base Split 3				
		1	2	3	5	10	1	2	3	5	10	1	2	3	5	10
Meta Learning	FSDetView [36]	56.9	64.8	64.0	69.6	69.8	33.0	53.1	56.4	67.7	70.0	42.7	53.8	58.3	66.1	67.4
	Meta R-CNN [34]	69.5	71.1	71.1	72.3	72.4	48.2	61.6	63.1	68.0	70.9	56.0	60.8	64.0	69.1	66.6
	VFA [48]	60.1	69.6	70.3	71.9	71.7	49.6	55.0	56.2	65.3	70.1	**65.4**	**68.3**	67.8	72.4	71.8
	FPD [35]	58.8	58.0	64.0	40.0	69.3	28.2	47.0	53.3	55.1	66.9	52.6	49.2	53.3	57.0	63.3
Fine Tuning	TFA w/cos [12]	68.6	71.8	73.2	72.4	72.6	46.6	58.3	61.0	67.8	69.0	52.4	64.7	64.2	71.1	74.3
	TFA w/fc [12]	**70.4**	**73.8**	**73.6**	**74.5**	**76.7**	44.6	62.7	**67.0**	67.6	68.9	47.4	57.6	**72.0**	70.7	74.6
	FSCE [40]	64.8	65.1	49.7	62.7	66.9	48.1	**65.5**	28.4	28.3	55.9	50.0	63.5	39.5	51.1	63.3
	DCFS [41]	62.8	67.7	67.1	68.5	72.1	36.3	59.5	58.5	65.4	70.5	40.0	59.4	64.8	68.1	71.0
	DeFRCN [38]	64.0	66.2	64.5	68.1	72.2	41.0	53.3	57.9	65.4	70.8	40.1	61.5	63.1	69.8	71.1
	HTRPN [37]	43.8	60.5	60.7	49.5	67.4	19.5	31.1	34.4	39.7	45.4	37.2	45.6	56.8	43.7	69.2
	FSNA [42]	69.4	70.6	69.5	73.7	72.5	28.2	61.5	62.9	66.1	70.6	54.9	65.5	69.6	71.6	74.8
	FSCE+SMLe [44]	61.1	67.1	66.1	57.3	74.0	36.2	55.6	56.5	67.3	69.8	52.5	60.7	60.9	69.0	73.8
	AGCM+SMLe [44]	66.3	70.2	70.4	73.4	75.2	48.4	61.2	61.4	**71.7**	**74.2**	58.0	66.1	68.4	**73.3**	**76.7**
	G-FSRD(Ours)	**78.9**	**78.8**	**79.1**	**79.0**	**78.9**	**80.6**	**80.4**	**80.4**	**80.5**	**80.4**	**78.7**	**78.9**	**76.6**	**78.3**	**76.7**

G-FSRD has better detection performance across many categories, as its area under the PR curves is consistently greater than TFA w/fc and HTRPN. Figure 9 also shows RoI activation heatmaps, which we used to evaluate G-FSRD and TFA's detection ability. In difficult situations like dim lighting and busy backdrops, G-FSRD demonstrates sharper edges and more concentrated responses (Fig. 9(a1)-(c3)). In contrast to G-FSRD's pinpoint accuracy, TFA's scattered and incorrect activations are seen for objects like Rail wayStraight and Helmet [Fig. 9(a2)-(e2)]. By achieving more precise and wider activation in the multiobject environment [Fig. 9(f)], G-FSRD demonstrates better robustness and semantic awareness. And to prove it even further, the categorization performance of

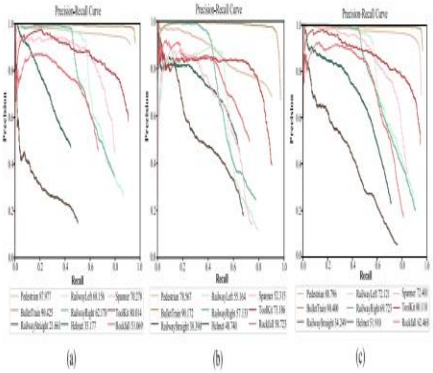


Fig. 8. Precision–recall curves of three methods on the ten-shot Split 1 of the FSRI2024 dataset. (a) TFA w/fc, (b) HTRPN, and (c) proposed G-FSRD. Each curve represents one intrusion category. Note: the HTRPN curve was generated from a retrained model due to checkpoint loss; its mAP_{50} (≈61.4) may slightly differ from Table II (61.1).

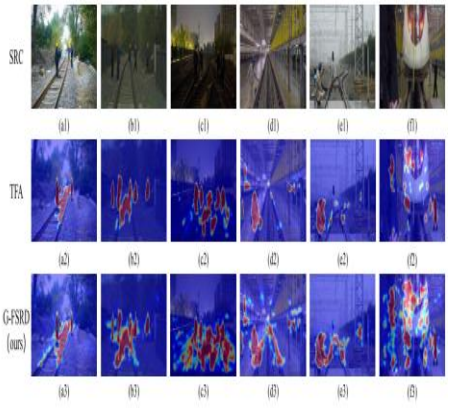


Fig. 9. Feature activation comparisons under typical railway scenarios. (a1)–(f1): input images (SRC); (a2)–(f2): heatmaps from TFA; (a3)–(f3): heatmaps from G-FSRD. Compared to TFA, G-FSRD shows more focused activations, clearer object boundaries, and better foreground-background separation.

TABLE VI

INFERENCE SPEED (FPS) ON DIFFERENT PLATFORMS UNDER VARIOUS PRECISIONS

Deployment Device	PyTorch	FP32	FP16	INT8
GTX 1080Ti	10.6	28.4	28.7	**57.1**
Jetson Xavier NX	1.47	2.6	10.0	**17.6**

TABLE VII

COMPARISON OF MODEL COMPLEXITY AND TRAINING RESOURCE CONSUMPTION BETWEEN G-FSRD AND TFA

Model	Parameter (M)	Memory (GB)	Iterations
TFA	60.3	22.47	11000
G-FSRD	62.5 (↑3.7%)	29.82 (↑32.7%)	4399 (↓60.0%)

We used G-FSRD to extract 100 RoI features for every class and then visualized them using t-SNE. Our approach has great clustering capabilities, as seen by this graphic. The baseline TFA [12] muddles classes 2-4, as shown in Fig. 10, as parts of classes 2 and 3 overlap with class 4. On the other hand, our G-FSRD neatly divides classes 2-4. Looking at the other classes, it's clear that G-FSRD has a more compact clustering effect than TFA [12]. The results show that G-FSRD outperforms the other approaches on FSRI2024. 3) Resource and Latency Analysis: Based on the FSRI2024 test set, Table V shows the average inference time per picture for each approach. Please be informed that the network requires an image size of 800 × 800 as input. FSCE [40] and HTRPN [37] were able to attain an inference time of 100 ms despite their more complicated structures, but

DeFRCN [38] had an inference time of 289 ms because of the extra classification module prototype calibration block (PCB). Meta RCNN [34] attained an inference speed of 91.4 ms based on meta-learning. Our technique ranked third with a speed of 95.6 ms, behind Meta RCNN [34] and TFA [12]. Nevertheless, the detection

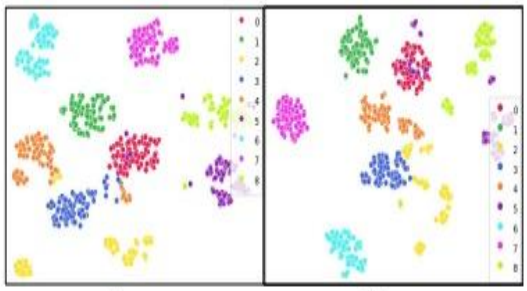


Fig. 10. t-SNE visualization of object embeddings in the feature space, obtained on FSRI2024 Split 1 under the ten-shot configuration. (a) TFA w/fc. (b) G-FSRD.

their level of accuracy is far lower than ours. In terms of speed and precision, our technique is superior. We ran inference tests on GTX 1080Ti and Jetson Xavier NX with FP32, FP16, and INT8 precision settings to assess G-FSRD's deployment viability. You may see the results in Table VI. By achieving 28.7 FPS (FP16) and 57.1 FPS (INT8) on 1080Ti, G-FSRD satisfies the demands of real-time applications. It achieves frame rates of 10.0 (FP16) and 17.6 (INT8) on Jetson NX. Regardless of

TABLE V
INFERENCE TIME IN MILLISECONDS ON THE FSRI2024 TEST SET

Method	Meta RCNN [34]	TFA w/fc [12]	DeFRCN [38]	FSCE [40]	HTRPN [37]	G-FSRD
Inf/ms	91.4	88.4	289.0	101.7	105.6	95.6

Although they fall short of the rigorous real-time requirement, the model has promise for enhancement by means of optimization of deployment or lowering of resolution. The fact that several train-mounted platforms now have processing capabilities on par with 1080Ti makes G-FSRD an excellent choice for host-side deployment. As shown in Table VII, we compared G-FSRD to the baseline TFA in terms of resource utilization. The results showed that G-FSRD enhanced efficiency with low overhead, with a 3.7% increase in parameter count, a 32.7% rise in memory usage, and a 60% reduction in training iterations. In conclusion, G-FSRD has great practical applicability and flexibility to embedded settings, while also striking a balance between architectural design and deployment efficiency.

C. Ablation Study and Visualization

First, we ran ablation tests on FSRI2024 in Split 1 with five- and ten-shot settings to make sure the suggested module would work. The results are in Table VIII. The same hyperparameters were used to train all the models unless otherwise stated. We started by reducing the performance of new classes after fine-tuning the baseline TFA's RPN and RoI independently [12]. But with the right adjustments, this outcome may be much better. The performance of the new classes was enhanced when we fine-tuned RPN and RoI together. Separately improving RPN and RoI also improved the performance of new classes as the number of samples rose. Together, fine-tuning RPN and RoI has improved novel class performance by increasing the number of foreground ideas used in future training. Also, we found that the accuracy of detecting base classes dropped significantly when we fine-tuned RPN and RoI Head. But in G-FSRD, the dual-detectors keep the detection jobs of base and novel classes distinct, stabilizing the base detector during fine-tuning and avoiding the reduction in performance of base classes, thus this problem doesn't impair the detector's performance. Using dual-detectors helped the basic courses maintain or even enhance performance, while the novel classes saw a dramatic improvement. We concluded by incorporating contrastive learning into the new sessions to further improve their performance. The strategy improved the performance of both base classes and new classes independently by increasing intraclass similarity and interclass dissimilarity. We contrast the visual outcomes of models with and without this dual-detector architecture to eloquently illustrate its efficacy. Figure 11 shows that the dual-detector improves the capacity to generalize to new classes while simultaneously reducing the risk of catastrophic forgetting of base classes. To be more precise, the absence of the dual-detector causes noticeable problems with pedestrian detection in the first two photos (as shown in Figure 11(a)) and, in the third image, in poor lighting. Furthermore, the model has trouble detecting little items like spanners and

helmets; for example, it misidentifies the spanner in the third picture and the helmet in the first. The dual-detector, on the other hand, makes it possible to detect them. In general, the dual-detector improves the model's flexibility in difficult railway intrusion settings and consistently detects typical base intrusions, while also detecting unusual unique intrusions to a far greater extent. In Table IX, we can see that we changed τ from 0.07 to 0.5 under contrastive dimensions Ds from 128 to 256 in order to assess the impact of temperature τ on the CLNC module. Poor feature separation or unstable gradients are probably to blame for the performance degradation seen at extreme levels (0.07 or 0.5). Remarkably, for Ds = 128, the best nAP50 (49.6) is achieved with τ = 0.2, which consistently produces high results across all settings. The default value of τ is set to 0.2, which is frequently utilized in previous research [40], [45], [50]. 2) In order to make G-FSRD's detection performance easier to grasp, Fig. 12 shows the visual results of G-FSRD, TFA w/fc

[12], and HTRPN [37]. Suburban settings, train station arrival, and turning are all part of the detection scenarios. distinct techniques are represented by each row, and the detection results of distinct Splits are shown by each column. The experimental findings show that G-FSRD can identify frequent base intrusions with high accuracy and identifies uncommon new intrusions successfully with little training data. This greatly improves its generalizability in railway intrusion detection. On the other hand, TFA w/fc [12] shows very little catastrophic forgetting of common base intrusions but isn't good enough at generalizing to uncommon novel intrusions, whereas HTRPN [37] is good at generalizing to uncommon novel intrusions but has catastrophic forgetting of common base intrusions. Both TFA w/fc [12] and HTRPN [37] identified a pedestrian in Split 1, however TFA only picked up the one on the right side of the first picture, while HTRPN only picked up the one on the far left of the second image. In this case, pedestrians are part of the common base incursions. Alternatively, G-FSRD was able to provide complete coverage with zero missed detections. During Split 2, while identifying obscured or tiny items (like Rockfall),

TABLE VIII
ABLATION STUDY FOR KEY COMPONENTS PROPOSED IN G-FSRD

Method	Refinement		Double Detector	Contrastive Learning within Novel Classes	5-shot			10-shot		
	RPN	RoI			nAP	bAP	mAP	nAP	bAP	mAP
TFA/fc	×	×	-	-	30.40	74.48	59.78	35.59	76.66	62.97
G-FSRD (Ours)	√	×	×	×	28.62	63.72	52.02	37.56	70.28	59.37
	×	√	×	×	29.35	58.96	49.09	40.34	67.89	58.70
	√	√	×	×	33.81	60.20	51.40	45.74	68.57	60.96
	√	√	√	×	39.55	**79.38**	66.16	48.35	**79.03**	68.81
	√	√	√	√	**40.77**	79.01	**66.26**	**49.64**	78.95	**69.18**

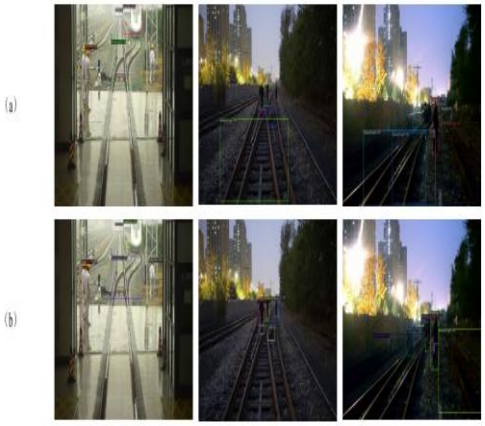


Fig. 11. Visual comparison of detection results: with and without the dual-detector. (a) w/o DD and (b) w DD.

TABLE IX
SENSITIVITY ANALYSIS OF THE TEMPERATURE PARAMETER τ UNDER DIFFERENT CONTRASTIVE EMBEDDING DIMENSIONS

Contrast Head Dimension	Temperature (τ)		
	0.07	0.2	0.5
D_c=128	48.4	**49.6**	47.4
D_c=256	49.1	48.3	46.9

There were notable restrictions with TFA w/fc [12] and HTRPN [37], particularly in low-light conditions, while using Helmet and ToolKit. When looking at the initial picture of Split 2, G-FSRD recognized all incursions and produced better visualizations than TFA w/fc [12] and HTRPN [37], which only spotted one RailwayStraight and two pedestrians, one rockfall, and one toolkit on the ground, respectively. Also, in Split 2's second picture, G-FSRD spotted all incursions, while TFA w/fc [12] and HTRPN [37] missed distant, Rockfall, Helmet, and ToolKit. When

INTERNATIONAL JOURNAL OF APPLIED SCIENCE ENGINEERING AND MANAGEMENT

ISSN:2454-9940 www.ijsem.org

Vol 20, Issuse.2 April 2026

pitted against TFA w/fc [12] and HTRPN [37], G-FSRD showed better confidence ratings and more precise bounding boxes in Split 3. For example, in the first picture, the BulletTrain on the far right was not picked up by TFA w/fc [12], and HTRPN [37] did not pick it up on either the far left or the far right. On the other hand, G-FSRD was able to detect everything. Further demonstrating its superiority, G-FSRD displayed greater confidence ratings in the second image as well. In addition, the visualization tests show that G-FSRD can overcome the issue of catastrophic for getting in frequent base intrusions and has remarkable generalization power in uncommon unique intrusion detection. D. Findings about Pascal VOC The Pascal VOC dataset was used for our generalization studies. In accordance with the TFA standard methodology [12], our technique, like with all the others in Table X, uses the Pascal VOC dataset with the identical class and data split parameters. Each of the three divisions has fifteen base classes and five innovative classes, for a total of twenty. Each class was trained with K=1, 2, 3, 5, or 10 samples during the fine-tuning step. For the sake of uniformity and fair comparison, we used the VOC2007 test set for evaluation and the VOC2007 + 12 training sets for training. Red indicates the top mAP50 values, while blue shows the second-best, as given in Table X. Overall, our model outperforms the competition; it ranks first in 12 assessment settings, second in 2, and third in 1 across all data Splits, according to these findings. This shows that our G-FSRD has strong generalizability capabilities, since it does well on both the FSRI2024 and VOC datasets. The effectiveness of fine-tuning strategies in reducing catastrophic forgetting is supported by experimental data showing that, in comparison to meta-learning-based methods, they often produce higher mAP50 scores. This provides further evidence that our G-FSRD, which is focused on fine-tuning, is the best

Fig. 12. Visualization of results from various models under the ten-shot configuration, where our G-FSRD demonstrates a notable improvement in detection performance.

TABLE X

RESULTS OF FSOD METHODS ON THE PASCAL VOC (07 + 12) ALL CLASSES $D_{base} \cup D_{novel}$ (MAP50) ACROSS THREE SPLITS. THE BEST AND SECOND-BEST RESULTS ARE MARKED IN RED/BLUE, RESPECTIVELY. † INDICATES THAT THE EXPERIMENTAL RESULTS FROM OFFICIAL PAPERS

Type	Method/Shots	All Split 1					All Split 2					All Split 3					Avg
		1	2	3	5	10	1	2	3	5	10	1	2	3	5	10	
Meta Learning	FSDetView† [36]	36.4	40.3	40.1	50.0	55.3	36.3	43.7	41.6	45.8	54.1	37.0	39.5	40.8	50.7	54.8	44.4
	Meta R-CNN† [34]	17.5	30.5	36.2	49.3	55.6	19.4	33.2	34.8	44.4	53.9	20.3	31.0	41.2	48.0	55.1	38.0
	VFA† [48]	47.4	54.4	58.5	64.5	66.5	33.7	38.2	43.5	48.3	52.4	43.8	48.9	53.3	58.1	60.0	51.4
	FSRW† [33]	53.5	50.2	55.3	56.0	59.5	55.1	54.2	55.2	57.5	58.9	54.2	53.5	54.7	58.6	57.6	55.6
	DCNet† [51]	33.9	37.4	43.7	51.1	59.6	23.2	24.8	30.6	36.7	46.6	32.3	34.9	39.7	42.6	50.7	39.2
Fine Tuning	TFA w/cos† [12]	**69.7**	**68.2**	**70.5**	**73.4**	**72.8**	**64.7**	**66.3**	**67.7**	**68.3**	68.3	**67.8**	**68.9**	**70.8**	**72.3**	**72.2**	**69.5**
	TFA w/fc† [12]	**69.3**	66.9	70.3	**73.4**	**73.2**	**64.7**	**66.3**	**67.7**	**68.3**	**68.7**	**67.8**	**68.9**	**70.8**	**72.3**	**72.2**	**69.5**
	FSCE† [40]	32.9	44.0	46.8	52.9	59.7	23.7	30.6	38.4	43.0	48.5	22.6	33.4	39.5	47.3	54.0	41.2
	MSPR† [39]	56.8	60.4	62.8	66.1	69.0	53.1	57.6	62.8	64.2	66.3	55.2	59.8	62.7	66.9	67.7	62.1
	DCFS† [41]	45.8	59.1	62.1	66.8	68.0	31.8	41.7	46.6	50.3	53.7	39.6	52.1	56.3	60.3	63.3	53.2
	DeFRCN† [38]	40.2	53.6	58.2	63.6	66.5	29.5	39.7	43.4	48.1	62.8	35.0	38.3	52.9	57.7	60.8	49.4
	G-FSRD(Ours)	**69.3**	**69.8**	**71.0**	**72.4**	72.4	**66.6**	**68.0**	**71.0**	**71.1**	**71.1**	**69.4**	**70.5**	**71.2**	**73.1**	**72.6**	**70.6**

answer to the problem of obtaining accurate, non-forgetting detection of railway intrusions.

V. CONCLUSION

In this research, we provide G-FSRD, a new FSOD approach that uses CLNCs, dual-detectors, and an effective fine-tuning framework to efficiently identify uncommon unique incursions on rail ways with small samples and prevent catastrophic forgetting of frequent base intrusions. In particular, the dual-detector ensures the preservation of rich information from base classes and mitigates catastrophic forgetting by decoupling the detection duties of new classes. The detection performance on new classes is improved by CNLCs because they increase intraclass compactness and interclass separability. The model's total detection performance is further optimized by an efficient fine-tuning framework, which allows each module to function in tandem with the others. Results from the extensive studies show that G-FSRD outperforms SOTA in detecting new classes and base classes, and it is very competitive in detecting overall detection. It is clear that G-FSRD has several benefits over the current FSOD approach. In addition, visualization results demonstrate its strong capacity to generalize, allowing for precise identification of railway intrusions in a variety of difficult environmental scenarios. Finally, the suggested approach offers a gradual and efficient detection framework for railway intrusion detection, which can properly identify both frequent and uncommon intrusions. The goal of future studies is to find ways to boost the detection accuracy of new classes without sacrificing performance on existing ones. In particular, we will go into more efficient methods of feature adaptation and sophisticated approaches to improving class separation. In complicated application settings, there are still some unresolved challenges, despite the encouraging outcomes. Before anything else, we don't yet have comprehensive validation of the model's robustness in severe situations, such blizzards or strong backlighting, even if FSRI2024 covers a number of scenarios. We want to do this by building a subset with clearly annotated incursion categories under extreme weather for the purpose of systematic assessment. Secondly, the model reaches 57.1 FPS on a GTX 1080Ti, but it only manages 17.1 FPS on Jetson NX, so there's still room for improvement in terms of real-time performance on edge devices. Consequently, in order to enhance efficiency and the practicality of deployment, we will look at dynamic inference techniques, pruning, and model compression.

REFERENCES

[1] X. Wu, W. Lian, M. Zhou, W. Bai, M. Yang, and H. Dong, "Critical spatial–temporal node identification for a high-speed railway network: A cascading delay perspective," *IEEE Trans. Netw. Sci. Eng.*, vol. 11, no. 1, pp. 823–833, Jan. 2024.

[2] T. Ye, Z. Zhao, S. Wang, F. Zhou, and X. Gao, "A stable lightweight and adaptive feature enhanced convolution neural network for efficient railway transit object detection," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 10, pp. 17952–17965, Oct. 2022.

[3] X. Gong, X. Chen, Z. Zhong, and W. Chen, "Enhanced few-shot learning for intrusion detection in railway video surveillance," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 8, pp. 11301–11313, Aug. 2022.

[4] T. Ye, J. Zhang, Z. Zhao, and F. Zhou, "Foreign body detection in rail transit based on a multi-mode feature-enhanced convolutional neural network," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 10, pp. 18051–18063, Oct. 2022.

[5] T. Ye, Z. Zheng, X. Li, Z. Zhao, and X.-Z. Gao, "An efficient few-shot object detection method for railway intrusion via fine-tune approach and contrastive learning," *IEEE Trans. Instrum. Meas.*, vol. 72, pp. 1–13, 2023.

[6] Z. Zhao, J. Kang, Z. Sun, T. Ye, and B. Wu, "A real-time and high-accuracy railway obstacle detection method using lightweight CNN and improved transformer," *Measurement*, vol. 238, Oct. 2024, Art. no. 115380.

[7] Y. Wu et al., "Rethinking classification and localization for object detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 10186–10195.

[8] O. Vinyals, C. Blundell, T. Lillicrap, K. Kavukcuoglu, and D. Wierstra, "Matching networks for one shot learning," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 29, Dec. 2016, pp. 3637–3645.

[9] Y. Wang, Q. Yang, L. Liu, and X. Zhang, "A cross-domain few-shot visual object tracker based on bidirectional adversary generation," *IEEE Sensors J.*, vol. 24, no. 12, pp. 19506–19516, Jun. 2024.

[10] Q. Liu, X. Zhang, Y. Liu, K. Huo, W. Jiang, and X. Li, "Multipolarization fusion few-shot HRRP target recognition based on metalearning framework," *IEEE Sensors J.*, vol. 21, no. 16, pp. 18085–18100, Aug. 2021.

[11] H. Zhang et al., "An extending interclass distance real-time network using positional orientation transformation for few-shot strip steel surface defect classification," *IEEE Sensors J.*, vol. 24, no. 24, pp. 42523–42537, Dec. 2024.

[12] X. Wang, T. E. Huang, T. Darrell, J. E. Gonzalez, and F. Yu, "Frustratingly simple few-shot object detection," 2020, *arXiv:2003.06957*.

[13] M. Grellert, B. Zatt, S. Bampi, and L. A. Da Silva Cruz, "Fast coding unit partition decision for HEVC using support vector machines," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 29, no. 6, pp. 1741–1753, Jun. 2019.

[14] C. Tastimur, M. Karakose, and E. Akin, "Image processing based level crossing detection and foreign objects recognition approach in railways," *Int. J. Appl. Math. Electron. Comput.*, no. 1, pp. 19–23, 2017.

[15] B. Guo, L. Yang, H. Shi, Y. Wang, and X. Xu, "High-speed railway clearance intrusion detection algorithm with fast background subtraction," *Chin. J. Sci. Instrum.*, vol. 37, no. 6, pp. 1371–1378, 2016.