Federated Adversarial Learning for Robust Autonomous Landing Runway Detection

Yi Li, Plamen Angelov, Zhengxin Yu, Alvaro Lopez Pellicer, Neeraj Suri

Computing and Communications, Lancaster University, UK

Abstract. As the development of deep learning techniques in autonomous landing systems continues to grow, one of the major challenges is trust and security in the face of possible adversarial attacks. In this paper, we propose a federated adversarial learning-based framework to detect landing runways using paired data comprising of clean local data and its adversarial version. Firstly, the local model is pre-trained on a large-scale lane detection dataset. Then, instead of exploiting large instance-adaptive models, we resort to a parameter-efficient fine-tuning method known as scale and shift deep features (SSF), upon the pre-trained model. Secondly, in each SSF layer, distributions of clean local data and its adversarial version are disentangled for accurate statistics estimation. To the best of our knowledge, this marks the first instance of federated learning work that address the adversarial sample problem in landing runway detection. Our experimental evaluations over both synthesis and real images of Landing Approach Runway Detection (LARD) dataset consistently demonstrate good performance of the proposed federated adversarial learning and robust to adversarial attacks.

1 Introduction

An unmanned aerial vehicle (UAV), commonly known as a drone, refers to an aircraft operated without a human pilot on board [1]. Drones find applications in various fields, such as photography, research, surveillance, defense, and space exploration. Enhancing the autonomy of aircraft is crucial as it reduces the cognitive load on pilots, ensuring safety in civil aviation [2]. Despite these advancements, unmanned aerial vehicles face challenges during the approach and landing phases. Recent progress in computer vision and embedded hardware platforms has positioned vision-based algorithms as an efficient direction for guiding and navigating during the landing stage. A vision-based landing system must be capable of detecting runways, from a distance to close proximity, in high-resolution images. The goal of Autonomous Landing Runway Detection (ALRD) is to identify and determine a suitable runway for an aircraft to land at an airport [3].

While there is considerable practical and commercial interest in autonomous landing systems within the aerospace field, there is currently a shortage of open-source datasets containing aerial images. To tackle this gap, a recent introduction of the Landing Approach Runway Detection (LARD) dataset [4] aims to provide a collection of high-quality aerial images specifically designed for the task of runway detection during approach and landing phases. The dataset primarily comprises images generated using conventional landing trajectories, where the possible positions and orientation of the aircraft during landing are defined within a generic landing approach cone.

In recent times, significant advancements in deep learning techniques have greatly enhanced their application in vision-based ALRD, resulting in commendable performance gains [5][3][6]. Typically, a neural network is trained using an extensive dataset of vision-based landing data to predict runways at varying distances in images captured by UAV cameras. However, the vulnerability to adversarial attacks in deep learning techniques poses a considerable risk in real-world ALRD applications. Adversarial attacks in federated learning, especially at the local image level, can manifest as data poisoning, data tampering, and privacy attacks. External attackers, for instance, may inject poisoned data into the local training dataset [7]. This may involve introducing adversarial examples crafted to deceive the local model during training. Conversely, in traditional large-scale neural network-based federated learning [8][9], a computational cost issue arises where each local client is required to train an individual model. However, as clients share the same task and dataset, they can potentially leverage shared features and the majority of weights.

The contributions of this paper are summarized as follows:

• We propose an approach based on federated learning for ALRD against adversarial attacks. Firstly, we pre-train a neural network on a large-scale lane detection dataset [10]. Next, the network is fine-tuned with paired images, *i.e.*, clean and adversarial images in M clients.

• Differing from conventional large-scale model-based federated learning [8][9], the weights of the proposed pre-trained model are frozen. In each client, we scale and shift the deep features (SSF) to leverage the representation abilities of large-scale pre-training models to achieve good performance on downstream tasks by fine-tuning a few trainable parameters.

• Distributions of clean local data and its adversarial version are disentangled for accurate statistics estimation. Consequently, deep features of paired images are jointly learned at each layer of the local model for the model to learn downstream information from adversarial images.

2 Related Works

2.1 Deep Learning for ALRD

The development of methods for detecting landing runways is pivotal for ensuring the security of autonomous aerial systems. These methods can be broadly categorized into two main approaches based on the data type: conventional image processing and deep learning-based image processing, video processing, multi-sensor fusion, and end-to-end learning. Firstly, conventional image processing involves techniques that manipulate and analyze images using traditional, rule-based algorithms such as edge detection [11], contour analysis [12], and color-based segmentation [13]. Secondly, deep learning-based techniques [5][6] have been developed and applied in image processing to improve the accuracy of aerial image detection. For instance, Akbar et al. proposed a two-stage modular approach where the first stage classifies aerial images to detect runways [5]. In the second stage, the identified runways are localized using both conventional line detection algorithms and more recent deep learning models. Finally, in contrast to conventional feature learning methods, some recent deep learning techniques [14] learn a mapping from raw sensor inputs to runway detection without explicit feature engineering. While these methods achieve high runway detection accuracy over image- or video-based datasets, they face three significant challenges [3]: 1) variable groundtruth sizes over parameters, *e.g.*, along track distance and vertical path angle, 2) robustness to adversarial attacks, 3) low execution time and computational costs.

2.2 Adversarial Attacks

Recent studies have highlighted the susceptibility of trained neural networks to compromise through adversarial samples, even with imperceptible perturbations that evade human detection [15]. This raises significant safety concerns regarding the deployment of such networks in real-world applications, including critical domains like autonomous driving and clinical settings [16].

The threat model categorizes existing adversarial attacks into three types: white-box, gray-box, and black-box attacks, differing in the level of knowledge possessed by adversaries [17]. Within these threat models, various attack algorithms for generating adversarial samples have been proposed, including the Fast Gradient Sign Method (FGSM) [18], Projected Gradient Descent (PGD) [19], Semantic Similarity Attack on High-Frequency Components (SSAH) [20], Carlini & Wagner (CW) [21], DeepFool [22][23], Basic Iterative Method (BIM) [24], and Jacobian-based Saliency Map Attack [25].

2.3 Federated Learning

Federated learning is a distributed machine learning approach that enables multiple clients to train a model collaboratively by using their local data without sharing [26]. A key characteristic of federated learning is that the training process takes place locally on each device. Instead of sharing raw data, only model updates are exchanged among the devices throughout the training process [27]. This mechanism effectively reduce the risk of data exposure, making it a privacy-preserving approach for collaborative model training over devices [28].

Most existing federated learning methods achieve promising performance over a wide range of tasks. However, these methods have two limitations. Firstly, [29][30][31] have shown that local data can be vulnerable to adversarial attacks. If an adversary can inject malicious data during the training phase, they may subtly alter the model's behavior. This could lead to vulnerabilities that can be exploited during inference. Secondly, the number of parameters of pre-trained models is usually very large [32][33], and simply fine-tuning the full model undoubtedly yields a huge amount of communication cost in federated learning algorithms. These limitations are addressed by the proposed federated learning pipeline in this work.

3 Proposed Approach

3.1 Preliminaries

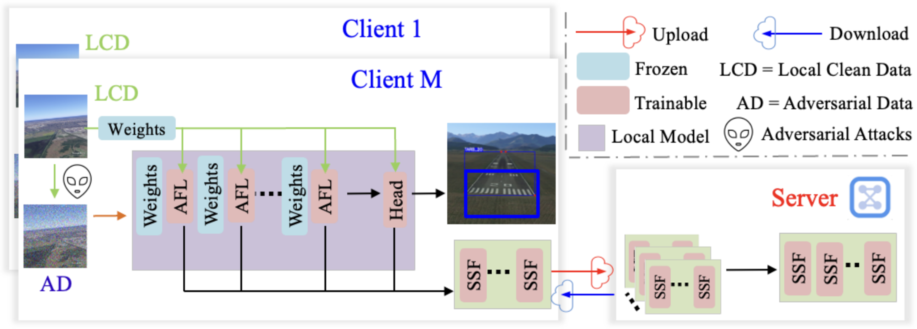

Fig. 1. Proposed pipeline of the federated learning-based landing runway detection method. The local models are initially pre-trained using lane detection datasets and subsequently fine-tuned with local landing runway detection datasets. The trained Scale and Shift Features (SSF) pools are then aggregated into the final model on the server.

In this section, we discuss the training of the proposed federated learning framework in subsections. The overall framework is presented in Fig. 1. In the federated learning framework, we assume there are M local clients, where each of them has their local dataset D_m containing clean samples C_m and adversarial samples A_m. The distributed paradigm FL aims to learn a central model with the parameter θ_c over the whole training data $\mathcal{D} = \{\mathcal{D}_1, \mathcal{D}_2, \ldots, \mathcal{D}_M\}$ using a central model without exchanging local private data. Formally, such a process can be expressed as:

$$\arg\min_{\theta_c} \mathcal{L}(\theta_c) = \sum_{m=1}^{M} \frac{|\mathcal{D}_m|}{|\mathcal{D}|} \mathcal{L}_m(\theta_c) \tag{1}$$

where the number of samples in D is presented as $|\mathcal{D}|$. $\mathcal{L}_m(\theta_c)$ is the empirical loss of the client m which can be expressed as:

$$\mathcal{L}_m(\theta_c) = \mathbf{E}_{(\mathbf{x},\mathbf{y})\in\mathcal{D}^m} \mathcal{L}_m(\mathbf{x};\theta_c) \tag{2}$$

where $\mathbf{x}$ denotes an image sample with the ground truth $\mathbf{y}$ of dataset D^m. $\mathcal{L}_m$ denotes the local loss term, *e.g.*, cross-entropy loss.

3.2 Pre-training

Recent landing runway detection techniques leverage deep learning, but the models in the existing literature are comparatively smaller than state-of-the-art models such as Vision Transformer (ViT) [34] and YOLO [35]. To address this, we propose fine-tuning a large-scale model pre-trained on a similar feature type-based dataset, such as lane detection datasets. This choice is motivated by the similarity between driving lanes and landing runways at the feature level. Specifically, we use a ViT-L/16 model [34] pre-trained on a road image dataset [10] to better extract features from our landing runway dataset. In the diagnostic experiment, detailed analysis of the chosen backbone, ViT-L/16, is provided.

3.3 Local Model Fine-tuning

The pre-trained model is then fine-tuned with airport aerial images from [4] to learn task-specific knowledge. However, there are two challenges in this stage. Firstly, pre-trained model usually has a large number of model parameters. Direct fine-tuning the full model consequently gives rise to significant communication costs between the server and the client. Secondly, recent studies show that adversarial attacks potentially poison the local data. This causes wrong predictions in autonomous landing systems. Therefore, we aim to mitigate these two challenges by using SSF and adversarial feature learning, respectively.

Scale and Shift Deep Features To efficiently fine-tune each local model, we scale and shift the deep features (SSF) the pre-trained model in the local training phase and merge them into the original pre-trained model weights by reparameterization in the inference phase. Given a pre-trained model with parameter θ, we send the model to each local client and the define the parameter as θ_m in the m-th client. In the fine-tuning stage, the model parameters of SSF can be represented as $\theta_m = \{\gamma_m, \beta_m, h_m, \theta_m\}$, where $\gamma_m \in R^D$ and $\beta_m \in R^D$ are scale and shift factors, respectively [36]. h_m is the parameter of the classification head. We break the weights of local models based on operations [36], *e.g.*, multi-head self-attention (MSA), MLP and BN, etc. Then, we remodulate features by inserting SSF with γ_m and β_m factors after these operations. It is highlighted that the pre-trained weights are kept frozen, and only the SSF and classification head are kept updated. Therefore, we define parameter ϕ_m and ϕ_c as the combination of trainable $\{\gamma, \beta, h\}$ in the m-th local model and central model, respectively. ϕ_c can be updated with Eq. (1) as:

$$\phi_c = \arg\min_{\phi\text{s}} \mathcal{L}(\phi_m) = \sum_{m=1}^{M} \frac{|\mathcal{D}_m|}{|\mathcal{D}|} \mathcal{L}_m(\phi_m) \tag{3}$$

where ϕs are the set of SSF, *i.e.*, $\phi_1, \phi_2, ...\phi_m$.

In each client, we updates ϕ_m in the r-th communication rounds between the server and local clients as:

$$\phi_m^{r,e+1} \leftarrow \phi_m^{r,e} - \eta_m \nabla \mathcal{L}_m \left(\mathbf{x}^m, \mathbf{a}^m, ; \phi_m^{r,e}\right) \tag{4}$$

where e is the index of local updates and η_m is the learning rate. Once the local model training is accomplished, ϕs can then be merged into θ_m to obtain the updated model parameter θ'_m. Besides, the server performs aggregation every communication round by receiving the updated parameters of local clients after the local updates within each round. Formally, we have

$$\phi_c^{e+1} \leftarrow \sum_{m=1}^{M} \frac{|\mathcal{D}^m|}{|\mathcal{D}|} \phi_m^e \tag{5}$$

Similarly, once all the communication rounds are accomplished, ϕ_cs can then be merged into θ_c to obtain a robust central model parameter without disclosing any local data.

Adversarial Feature Learning In recent studies [37][38][39], clean and adversarial images have different underlying distributions because the adversarial images essentially involve a two-component mixture distribution. Therefore, in the proposed adversarial feature learning, we aim to disentangle features from the clean and adversarial images to enhance the global feature representation and suppress the adversarial attacks. To achieve that, we generate adversarial images from the clean images by using the attack algorithms, *e.g.*, FGSM. Then, these paired clean and adversarial image samples are fed into the proposed adversarial feature learning (AFL) block. Due to different underlying distributions of clean and adversarial images, different from conventional adversarial image learning techniques [40], we exploit different normalization techniques for clean and adversarial images to guarantee its normalization statistics are exclusively preformed on the adversarial images. Particularly, the batch normalization (BN) [41] and random normalization aggregation (RNA) [42] are empirically used for clean and adversarial images, respectively. The support experiments for the chosen experiment configuration will be provided in Section 4.5.

The loss between clean and adversarial images at the m-th client is defined as:

$$\mathcal{L}_m = \frac{1}{N} \sum_{i=1}^{N} \|(\gamma_{\text{cl}} \odot \mathbf{x}_n^m + \beta_{\text{cl}}) - (\gamma_{\text{adv}} \odot \mathbf{a}_n^m + \beta_{\text{adv}}\|_2^2 \tag{6}$$

where N is the number of samples in a local client. The clean and adversarial samples are denoted as $\mathbf{x}_n^m$ and $\mathbf{a}_n^m$, respectively. Except BN and RNA, clean and adversarial sample parameters γ_{cl}, β_{cl}, γ_{adv}, and β_{adv} for convolutional and other layers are jointly optimized for both adversarial examples and clean images. Specifically, the AFL with an RNA helps to learn the features by keeping separate BNs to features that belong to different domains.

3.4 Central Model Update

When the training is accomplished, we can re-parameterize the SSF by merging it into the original parameter space (*i.e.*, model weight θ). As a result, federated

SSF is not only efficient in terms of communication costs, but also does not introduce any extra parameters during the inference phase. The algorithm of the proposed federated adversarial learning framework is presented in Algorithm 1.

Algorithm 1 Federated adversarial learning

```
1: **Input:** Local datasets of $M$ clients: $\mathcal{D} = \{\mathcal{D}_1, \mathcal{D}_2, \ldots, \mathcal{D}_M\}$, clean samples $\mathbf{x}^m$, adversarial samples $\mathbf{a}^m$, maximum local update $E$, communication rounds $R$, learning rate $\eta_m$, learnable parameters $\phi_m^{r,e}$
2: **Output:** The final central model $\theta_c$
3: Initialize SSF
4: for $e = 1, ..., E$ do
5:     for $m = 1, ..., M$ in parallel do
6:         Download from the central model to local models: $\phi_m^{r,e} \leftarrow \phi_c^e$
7:         Local model updates in clients: $\phi_m^{r,e+1} \leftarrow \phi_m^{r,e} - \eta_m \nabla \mathcal{L}_m \left(\mathbf{x}^m, \mathbf{a}^m; \phi_m^{r,e}\right)$
8:     end for
9:     Update local models to the central model: $\phi_c^{e+1} \leftarrow \sum_{m=1}^{M} \frac{|\mathcal{D}_m|}{|\mathcal{D}|} \phi_m^e$
10:    $\theta_c^{e+1} = \phi_c^{e+1}$
11: end for
12: SSF Aggregation: $\theta_c = \text{Agg}(\theta_c^1, \theta_c^2, ..., \theta_c^E)$
```

4 Experimental Results

4.1 Dataset and Attacks

The tvtLANE dataset [10] contains 19,383 image sequences for lane detection, and 39,460 frames of them are labeled. In this work, the model is pre-trained with randomly selected 35,000 images from the dataset. We further fine-tune the local models on synthetic and real runways from the LARD dataset [4]. The LARD dataset contains 14,433 training images of resolution 2448×2648, taken from 32 runways in 16 different airports in total. In the training stage, we set number of clients to 5, and randomly select 2,800 for each client. Then, we construct the validation set with the rest 433 images. In the test stage, we evaluate the central server with 2,221 synthetic images taken from 79 runways in 40 different airports and 103 hand-labeled pictures from real landing footage on 38 runways in 36 different airports.

We select aforementioned attacks in Section II because they are robust to novel adversarial attack recovery techniques [17][43]. The adversarial images in the training and test stages are generated with the same attack algorithm.

4.2 Competitors and Performance Measure

In this paper, the proposed method is evaluated and compared to state-of-the-art competitor models. Firstly, we select four landing runway detection techniques, including long short-term memory (LSTM) [44], Line Segment Detector (LSD)

[5], Runway Detection Systems (RDS) [6], Complex Cross-Residual Network (CS-ResNet) [45] as the original implementations in the literature but with same data as the proposed method. Secondly, we use four federated learning frameworks, including Siloed Batch Normalization (SiloBN) + Adaptive Sharpness-Aware Minimization (ASAM) [46], federated optimization (FedProx) [47], federated averaging (FedAvg) [26], and federated local drift decoupling and correction (FedDC) [48] which are state-of-the-art in image processing tasks to confirm the proposed model is robust to adversarial attacks. For a fair comparison, we reproduce these models with same data as the proposed method. In the experiment, we calculate the average error between 6 predictions and ground truths (e.g., along track distance, vertical path angle, lateral path angle, yaw, pitch, and roll).

4.3 Model Configuration

We pre-train a Vit-L/16 model [34] as the backbone by using PaddleSeg toolkit [49]. The backbone study is presented later in diagnostic experiments. The model is trained by using the SGD optimizer with a weight decay of 0.0001, a momentum of 0.9, and a batch size of 256. We train the model for 300 epochs. The initial learning rate is 0.03, and is multiplied by 0.1 at 500 and 1000 epochs.

After the pre-training, we only fine-tune the SSF layers and freeze the other weights of the Vit-L/16 model. In particular, we fine-tune the model for 100 epochs. All experiments are run on the High End Computing (HEC) Cluster with Tesla V100 GPUs.

4.4 Comparison to State-of-the-Arts

We conduct two experiments in this section. In the first experiment, we assume the federated learning perfectly protect the data privacy. Therefore, the central model is evaluated with *clean* samples of LARD [4]. Table 1 shows the results, each of them is the average of 2,221 synthesis images or 103 real images. In the second experiment, we evaluate the performance over *adversarial* data and present the results in Fig. 2(2).

Table 1 shows that our federated adversarial learning improves results over landing runway detection models and federated algorithms. Our model is 3.3 points lower than CS-ResNet (19.5% vs. 22.8%). We conduct more experiments to confirm this point in Fig. 2. Adversarial images are generated from clean samples of LARD [4] for the model training and evaluation. Fig.2 (a) shows the results, each of them is the average of 15,547 (2,221 synthesis samples) synthesis images or 721 (103 real samples $\times$ 7 attack algorithms) real images.

From Fig.2(a), it can be observed that in all the evaluated models, the proposed model achieves 21.4% and 29.0% for synthesis and real images detection, respectively, which offers the best effectiveness. These observations are consistent with clean images. Moreover, comparing the results to Table 1, competitor models suffer a significant accuracy degradation with adversarial attacks, while the

Table 1. Detection error comparison with ALRD and FL (federated learning) algorithms on the **clean** samples of LARD dataset. Para. denotes the **trainable** parameters.

Method	FL	Para. (M)	Synthesis (%)	Real (%)
LSD [5]	✗	25.6	28.6	37.5
LSTM [44]	✗	0.5	28.0	35.8
RDS [6]	✗	38.9	25.2	34.1
CS-ResNet [45]	✗	58.2	22.8	31.9
FedAvg [26]	✓	1.7	29.5	37.6
FedProx [47]	✓	53.2	26.1	35.2
FedDC [48]	✓	11.2	25.0	35.0
SiloBN + ASAM [46]	✓	4.5	24.8	32.6
Ours	✓	7.4	**19.5**	**26.3**

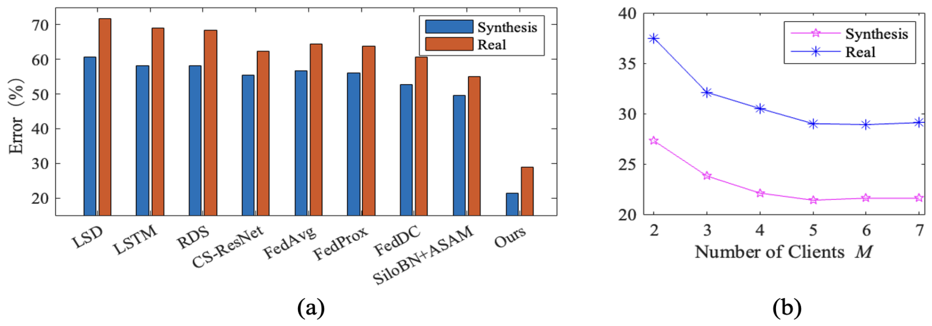

Fig. 2. Detection error comparison (a) to competitor models (b) against number of clients over LARD.

proposed models perform more robust because the global feature representation is enhanced to suppress the adversarial attacks.

4.5 Diagnostic Experiment

In this section, we first conduct experiments to validate several intriguing properties, *e.g.*, backbone. Then, we study the efficacy of our core ideas and essential pipeline design.

Number of Clients We conducted experiments to demonstrate the trade-off between performance improvement and network depth, specifically, varying the number of proposed PAA blocks. Fig. 2(b) presents these results, with each data point being an average of 15,547 (2,221 synthesis samples $\times$ 7 attack algorithms) synthesis images or 721 (103 real samples $\times$ 7 attack algorithms) real images.

Fig. 2(b) compares detection error against the number of clients on LARD. As Fig. 4 shows, detection errors start to decrease from $M = 2$, but performance is fairly stable for $5 \leq M \leq 7$. Therefore, the results indicate that $M = 5$ offers the best trade-off, validating the chosen implementation setting.

Backbone We implement the proposed federated learning framework with different backbones. The backbones are pre-trained on the tvtLANE dataset [10] and fine-tuned in clients. The experimental results are provided in Table 2. Each result of them is the average of 15,547 (2,221 synthesis samples × 7 attack algorithms) synthesis images or 721 (103 real samples × 7 attack algorithms) real images.

Table 2. Backbones.

	Para. (M)		Error (%)	
	Backbone	SSF	Synthesis	Real
ResNet50 [50]	25.6	-	28.5	36.8
ResNet101 [50]	42.8	-	28.1	35.6
ENet-B3 [51]	12.0	-	25.8	39.1
ENet-B5 [51]	30.6	-	24.7	38.0
ENet-B7 [51]	66.0	-	24.0	36.3
WRN-50-2 [52]	68.9	-	29.7	38.2
WRN-101-2 [52]	126.8	-	28.0	35.6
VGG16 [53]	138.3	-	31.4	42.9
ViT-S-16	22.0	3.96	26.6	35.8
ViT-B-16	86.5	4.62	23.0	33.1
ViT-L-16	307.1	7.39	**21.4**	**29.0**

Table 3. Normalization techniques.

		Adversarial				
		BN	LN	IN	GN	RNA
Clean	BN	27.5	31.9	38.2	36.8	**23.4**
	LN	32.6	34.7	40.6	31.2	33.3
	IN	35.9	41.2	40.5	30.7	29.8
	GN	30.4	29.8	35.7	29.1	31.8
	RNA	25.8	29.5	34.2	33.9	25.9

Table 2 compares the detection error against backbone networks on LARD. The results indicate that: 1) Although the ViT family requires massive network parameters, the SSF significantly reduces the parameters, which facilitates the training; 2) ViT-L-16 with SSF offers the best trade-off between the detection performance and computational cost, supporting the chosen experiment configuration.

Normalization Techniques A diagnostic experiment of normalization techniques including BN [41], layer normalization (LN) [54], instance normalization (IN) [55], group normalization [56], and RNA [42] for clean and adversarial images is conducted on the LARD dataset. Each result in Table 3 is an average of 16,268 experiments (2,221 synthesis samples × 7 attack algorithms + 103 real samples × 7 attack algorithms).

According to Table 3, BN and RNA are optimal for clean and adversarial image features, respectively. The detection error achieves 23.4% at the valley, which supports the experiment configuration.

Ablation Study In this section, we investigate the effectiveness of each contribution based on the LARD dataset. The ablation study is presented in Table 4 and the experimental setting is the same as Section 4.4. Each result is the average of 15,547 (2,221 synthesis samples × 7 attack algorithms) synthesis images or 721 (103 real samples × 7 attack algorithms) real images.

Table 4. Ablation study in the proposed method. afl and fl denote adversarial feature learning and federated learning, respectively.

Ablation Settings			Para. (M)	Synthesis (%)	Real (%)
AFL	FL	SSF			
✗	✗	✗	307.1	52.5	59.1
✓	✗	✗	307.1	31.6	38.3
✗	✓	✗	1535.5	44.2	50.7
✗	✗	✓	1.5	52.3	59.0
✗	✓	✓	7.39	43.9	48.5
✓	✗	✓	1.5	34.8	42.6
✓	✓	✗	1535.5	21.5	29.5
✓	✓	✓	7.39	**21.4**	**29.0**

Initially, we evaluate the effectiveness of AFL, which plays a pivotal role in learning desired features from adversarial images. AFL demonstrates its significant impact within the federated learning framework and SSF, resulting in a remarkable performance improvement from an initial error rate of 43.9% to 21.4%. This improvement can be attributed to the enhanced global feature representation provided by AFL, effectively suppressing adversarial attacks.

Moreover, the detection error experiences a significant reduction by exploiting federated learning framework (*i.e.*, 34.8% $\rightarrow$ 21.4%, synthesis images). The SSF aggregation captures diverse features learned from different local clients and data, empowering the central client to make more accurate predictions through the assimilation of rich features.

The final experiment in the ablation study involves the addition of SSF. Consequently, federated SSF not only prove to be efficient in terms of communication costs but also do not introduce any extra parameters during the test stage.

4.6 Visualizations

In this section, we present qualitative results demonstrating the landing runway detection of attacked image samples on LARD [4] in Fig. 3.

After comparing the reconstructed images with the original and attacked images, it can be observed that the detection boxes obtained via the proposed model with adversarial training provide more accurate descriptions of the landing runway areas. This observation further confirms the efficacy of the proposed method.

5 Conclusion

In this paper, we have proposed a federated adversarial learning framework as a simple yet effective alternative to conventional landing runway detection algorithms. To efficiently fine-tune the pre-trained local model on clients, we utilize a technique of shifting and scaling features with both clean and adversarial

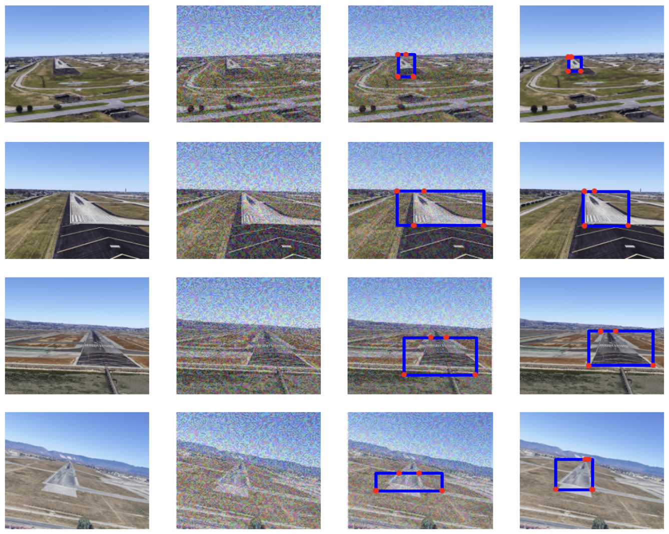

Fig. 3. Qualitative landing runway detection results on LARD. From left to right: original images, images attacked by FGSM, results of the central model without adversarial training, results of the central model.

samples. Subsequently, the SSF pools are aggregated into the central model. Our evaluation on LARD has demonstrated the high efficiency and effectiveness of the proposed model through the use of qualitative results and quantitative results which includes detection error comparison between ALRD and FL, different backbones, different normalisation techniques and lastly a thorough ablation study amongst others.

Acknowledgment

Research supported by the UKRI Trustworthy Autonomous Systems Node in Security/EPSRC Grant EP/V026763/1.

References

1. P. Wang, Z. Yang, X. Chen, and H. Xu, "A Transformer-based method for UAV-view geo-localization," *Proceedings of the International Conference on Artificial Neural Networks (ICANN)*, 2023.
2. X. Pang, N. Zhao, J. Tang, C. Wu, D. Niyato, and K. Wong, "IRS-assisted secure UAV transmission via joint trajectory and beamforming design," *IEEE Transactions on Communications*, vol. 70, no. 2, pp. 1140 – 1152, 2022.
3. B. Ajith, S. D. Adlinge, S. Dinesh, U. P. Rajeev, and E. S. Padmakumar, "Robust method to detect and track the runway during aircraft landing using colour segmentation and runway features," *Proceedings of International Conference on Trends in Electronics and Informatics (ICOEI)*, 2019.

4. M. Ducoffe, M. Carrere, L. Féliers, V. M. A. Gauffriau, C. Pagetti, and T. Sammour, "LARD - landing approach runway detection - dataset for vision based landing," *arXiv preprint arXiv: 2304.09938*, 2023.
5. J. Akbar, M. Shahzad, M. I. Malik, A. Ul-Hasan, and F. Shafait, "Runway detection and localization in aerial images using deep learning," *Proceedings of Digital Image Computing: Techniques and Applications (DICTA)*, 2019.
6. N. Drougard and M. Cassaro, "Implementation of runway detection systems using machine learning," *National Higher French Institute of Aeronautics and Space*, 2022.
7. Y. Li, P. Angelov, and N. Suri, "Adversarial attack detection via fuzzy predictions," *Proceedings of IEEE Symposium Series on Computational Intelligence (SSCI)*, 2023.
8. J. Tao, Z. Gao, and Z. Guo, "Training vision Transformers in federated learning with limited edge-device resources," *Electronics*, vol. 11, no. 17, p. 2638, 2022.
9. L. Qu, Y. Zhou, P. P. Liang, Y. Xia, F. Wang, E. Adeli, L. Fei-Fei, and D. Rubin, "Rethinking architecture design for tackling data heterogeneity in federated learning," *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.
10. Q. Zou, Q. D. H. Jiang, and, Y. Yue, L. Chen, and Q. Wang, "Robust lane detection from continuous driving scenes using deep neural networks," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 1, p. 41–54, 2020.
11. D. Garlin and V. Naidu, "Detection of airport runway edges using line detection techniques," *Core*, 2011.
12. Z. Hua, Z. Bian, and J. Li, "Airport detection-based cosaliency on remote sensing images," *Mathematical Problems in Engineering*, vol. 2021, no. 1, pp. 1 – 17, 2021.
13. D. Kordos, P. Krzaczkowski, and E. Zesławska, "Vision system measuring the position of an aircraft in relation to the runway during landing approach," *Mathematical Problems in Engineering*, vol. 23, no. 3, p. 1560, 2023.
14. R. A. Amit and C. K. Mohan, "A robust airport runway detection network based on R-CNN using remote sensing images," *IEEE Aerospace and Electronic Systems Magazine*, vol. 36, no. 11, pp. 4–20, 2021.
15. X. Ma, Y. Niu, L. Gu, Y. Wang, Y. Zhao, J. Bailey, and F. Lu, "Understanding adversarial attacks on deep learning based medical image analysis systems," *Pattern Recognition*, vol. 110, p. 107332, 2021.
16. X. Yuan, P. He, Q. Zhu, and X. Li, "Adversarial examples: attacks and defenses for deep learning," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 30, no. 9, pp. 2805 – 2824, 2019.
17. Y. Li, P. Angelov, and N. Suri, "Domain generalization and feature fusion for cross-domain imperceptible adversarial attack detection," *Proceedings of the International Joint Conference on Neural Networks (IJCNN)*, 2023.
18. I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," *Proceedings of International Conference on Learning Representations (ICLR)*, 2015.
19. A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, "Towards deep learning models resistant to adversarial attacks," *Proceedings of International Conference on Machine Learning (ICML)*, 2017.
20. C. Luo, Q. Lin, W. Xie, B. Wu, J. Xie, and L. Shen, "Frequency-driven imperceptible adversarial attack on semantic similarity," *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.
21. N. Carlini and D. Wagner, "Towards evaluating the robustness of neural networks," *IEEE Symposium on Security and Privacy*, 2017.

22. S.-M. Moosavi-Dezfooli, A. Fawzi, and P. Frossard, "Deepfool: a simple and accurate method to fool deep neural networks," *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.
23. A. L. Pellcier, Y. Li, and P. Angelov, "PUDD: Towards Robust Multi-modal Prototype-based Deepfake Detection," *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024.
24. A. Kurakin, I. Goodfellow, and S. Bengio, "Adversarial examples in the physical world," *arXiv preprint arXiv:1607.02533*, 2016.
25. N. Papernot, P. McDaniel, S. Jha, M. Fredrikson, Z. B. Celik, and A. Swami, "The Limitations of Deep Learning in Adversarial Settings," *IEEE Symposium on Security and Privacy*, 2016.
26. H. B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. Arcas, "Communication-efficient learning of deep networks from decentralized data," *International Conference on Artificial Intelligence and Statistics (ICAIS)*, 2017.
27. K. Bonawitz, P. Kairouz, B. McMahan, and D. Ramage, "Federated learning and privacy: building privacy-preserving systems for machine learning and data science on decentralized data," *ACM Queue*, vol. 19, no. 5, p. 87–114, 2021.
28. N. Truong, K. Sun, S. Wang, F. Guitton, and Y. Guo, "Privacy preservation in federated learning: an insightful survey from the GDPR perspective," *Computers and Security*, vol. 110, p. 102402, 2021.
29. A. K. Nair, E. D. Raj, and J. Sahoo, "A robust analysis of adversarial attacks on federated learning environments," *Computer Standards and Interfaces*, vol. 86, p. 103723, 2023.
30. L. Shi, Z. Chen, Y. Shi, G. Zhao, L. Wei, Y. Tao, and Y. Gao, "Data poisoning attacks on federated learning by using adversarial samples," *International Conference on Computer Engineering and Artificial Intelligence (ICCEAI)*, 2022.
31. S. Queyrut, V. Schiavoni, and P. Felber, "Mitigating adversarial attacks in federated learning with trusted execution environments," *International Conference on Distributed Computing Systems (ICDCS)*, 2023.
32. Y. Shu, Z. Kou, Z. Cao, J. Wang, and M. Long, "Zoo-tuning: adaptive transfer from a zoo of models," *Proceedings of International Conference on Machine Learning (ICML)*, 2021.
33. W. Kim, B. Son, and I. Kim, "Vilt: vision- and-language transformer without convolution or region supervision," *Proceedings of International Conference on Machine Learning (ICML)*, 2021.
34. A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, "An image is worth 16x16 words: Transformers for image recognition at scale," *Proceedings of International Conference on Learning Representations (ICLR)*, 2021.
35. Z. Ge, S. Liu, F. Wang, Z. Li, and J. Sun, "YOLOX: exceeding YOLO series in 2021," *arXiv preprint arXiv:2107.08430*, 2021.
36. D. Lian, D. Zhou, J. Feng, and X. Wang, "Scaling & shifting your features: a new baseline for efficient model tuning," *Proceedings of Advances in Neural Information Processing Systems (NeurIPS)*, 2022.
37. C. Xie, M. Tan, B. Gong, J. Wang, A. Yuille, and Q. V. Le, "Adversarial examples improve image recognition," *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.
38. Y. Li, P. Angelov, and N. Suri, "Rethinking self-supervised learning for cross-domain aversarial sample recovery," *Proceedings of the International Joint Conference on Neural Networks (IJCNN)*, 2024.

39. A. L. Pellcier, K. Giatgong, Y. Li, N. Suri, and P. Angelov, "UNICAD: A unified approach for attack detection, noise reduction and novel class identification," *Proceedings of the International Joint Conference on Neural Networks (IJCNN)*, 2024.
40. X. Shi, Y. Peng, Q. Chen, T. Keenan, A. T. Thavikulwat, S. Lee, Y. Tang, E. Y. Chew, R. M. Summers, and Z. Lu, "Robust convolutional neural networks against adversarial attacks on medical images," *Pattern Recognition*, vol. 132, p. 108923, 2022.
41. S. Ioffe and C. Szegedy, "Batch normalization: accelerating deep network training by reducing internal covariate shift," *Proceedings of International Conference on Machine Learning (ICML)*, 2015.
42. M. Dong, X. Chen, Y. Wang, and C. Xu, "Random normalization aggregation for adversarial defense," *Proceedings of Conference on Neural Information Processing Systems (NeurIPS)*, 2022.
43. C. Cintas, S. Speakman, V. Akinwande, W. Ogallo, K. Weldemariam, S. Sridharan1, and E. McFowland, "Detecting adversarial attacks via subset scanning of autoencoder activations and reconstruction error," *Proceedings of International Joint Conference on Artificial Intelligence (IJCAI)*, 2020.
44. X. Han, T. Zhou, Y. He, Y. Chen, R. Chen, and W. Zhou, "LSTM-based visibility detection for airport images in time series," *Chinese Control Conference (CCC)*, 2021.
45. P. Han, Y. Liu, and Z. Cheng, "Airport runway detection based on a combination of complex convolution and ResNet for PolSAR images," *SAR in Big Data Era (BIGSARDATA)*, 2021.
46. D. Caldarola, B. Caputo, and M. Ciccone, "Improving generalization in federated learning by seeking flat minima," *Proceedings of European Conference on Computer Vision (ECCV)*, 2022.
47. T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith, "Federated optimization in heterogeneous networks," *Proceedings of Machine Learning and Systems (MLSys)*, 2020.
48. L. Gao, H. Fu, L. Li, Y. Chen, M. Xu, and C.-Z. Xu, "FedDC: federated learning with non-iid data via local drift decoupling and correction," *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.
49. Y. Liu, L. Chu, G. Chen, Z. Wu, Z. Chen, B. Lai, and Y. Hao, "Paddleseg: a high-efficient development toolkit for image segmentation," *arXiv preprint arXiv: 2101.06175*, 2021.
50. K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.
51. M. Tan, R. Pang, and Q. V. Le, "Efficientdet: scalable and efficient object detection," *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.
52. S. Zagoruyko and N. Komodakis, "Wide residual networks," *The British Machine Vision Conference (BMVC)*, 2016.
53. K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *Proceedings of International Conference on Learning Representations (ICLR)*, 2015.
54. J. L. Ba, J. R. Kiros, and G. E. Hinton, "Layer normalization," *arXiv preprint arXiv: 1607.06450*, 2016.

55. D. Ulyanov, A. Vedaldi, and V. Lempitsky, "Instance normalization: the missing ingredient for fast stylization," *arXiv preprint arXiv: 1607.08022*, 2016.
56. Y. Wu and K. He, "Group normalization," *Processdings of European Conference on Computer Vision (ECCV)*, 2018.