VESPA: A General System for Vision-based Extrasensory Perception Anti-cheating in Online FPS Games

Shiwei Zhao, Jiaheng Qi, Zhipeng Hu, Han Yan, Runze Wu, Xudong Shen, Tangjie Lv, Changjie Fan
Fuxi AI Lab, NetEase Games, Hangzhou, China
{zhaoshiwei, qijiaheng, zphu, yanhan, wurunze1, hzshenxudong, hzlvtangjie, fanchangjie}@corp.netease.com

***Abstract*—Cheating is widespread in online games, particularly in competitive games like First-Person Shooter (FPS) games. One of the most common types of cheating is Extrasensory Perception (ESP), which involves illicitly obtaining visual information to gain an unfair advantage over normal players. To protect the gaming experience of legitimate players and the interests of game companies, there is an urgent need for anti-cheating applications. In this paper, we propose a general system for ESP anti-cheating in online FPS games, considering the business characteristics and industrial applications. We present a vision-based anti-cheating framework that incorporates both supervised and unsupervised solutions for comprehensive cheating detection. Based on this framework, we design and deploy a dual-audit human-in-the-loop system for industrial gaming anti-cheating applications. We evaluate our proposed framework from multiple online and offline perspectives and demonstrate its practical significance with superior performance.**

***Index Terms*—Extrasensory perception, anti-cheating system, online FPS games, industrial applications.**

I. Introduction

NEARLY 3.2 billion people worldwide play games and spend a combined total of $196.8 billion in 2022[1]. Online games make up the largest segment of the global game market, with online First-Person Shooter (FPS) games being widely favored for their competitive and playable nature. However, online FPS games suffer from cheating [1], [2]. This is because players have easy access to cheating techniques and are not always penalized severely even if caught. Recent research has shown that even a small percentage of cheaters can ruin the game experience for a large percentage of players: 6% of players cheat resulting in the chance of encountering at least one cheater being 42.7% in 5 vs. 5 matches (e.g., Valorant[2] or Counter-Strike: Global Offensive[3]) and 49.4% in 6 vs. 6 matches (e.g., Overwatch[4]) [2].

Cheating in online FPS games can be categorized into three types based on the gain obtained [2]–[5]. First, aimbots enhance or completely control a player's aim to target the enemy's position for more accurate shots. Second, movement cheating allows players to abuse game mechanics that increase their movement. As shown in Figure 1, we focus on the last major category of cheating, namely extrasensory perception (ESP), also known as visual hacks (or Wallhacks[5]), which renders illicit information (e.g., enemy positions, health, and equipment). Anti-cheating efforts typically rely on two main solutions: program-based and data-based solutions. The former is mainly executed on the client side and works much like anti-virus software to detect suspicious programs, with security and privacy concerns [6]. The latter is mainly deployed on the server side and identifies cheating players from normal players based on differences in various game data [1], [7]–[10]. However, such methods are mostly game-specific and thus lack generality and interpretability.

This paper aims to propose an efficient, accurate, and appli-

(a)

(b)

Fig. 1. ESP provides cheating players with more extra visual information in online FPS games.

Corresponding authors: Runze Wu and Xudong Shen.

[1]https://newzoo.com/resources/trend-reports/newzoo-global-games-market-report-2022-free-version

[2]https://playvalorant.com

[3]https://www.counter-strike.net

[4]https://overwatch.blizzard.com

[5]Some existing studies do not clearly distinguish between Wallhack and ESP and generally lump them into the same category. Technically, Wallhacks traditionally refer to plugins that render walls transparent or translucent and are therefore only a subset of ESP.

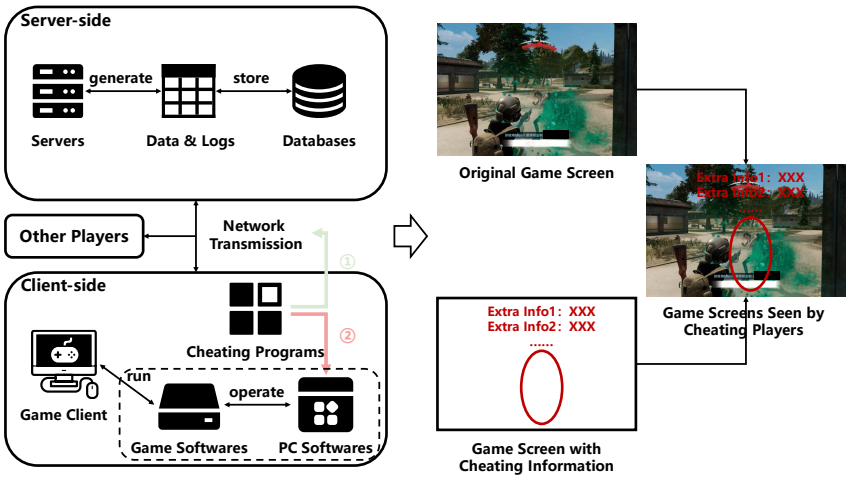


Fig. 2. The workflow of ESP cheating in online FPS games. Cheating programs can obtain cheating information by ① catching server-to-client packets and ② reading local memory.

cable systematic solution for vision-based ESP anti-cheating in industrial applications of online FPS games. To achieve these goals, we must address three key challenges:

Challenge 1: ESP anti-cheating solutions must consider a variety of factors. Current anti-cheating solutions are inadequate for ESP anti-cheating because they have issues with privacy protection, detection performance, and generality. A vision-based solution is a promising alternative because there is a significant visual difference between normal and cheating players that can serve as a useful feature and evidence [2], [11], [12]. And this solution does not require any additional personal privacy information and the image data used is secure and reliable through complete anonymization. More importantly, this solution can be easily generalized to different FPS games, saving a lot of labor and material costs for large game companies. Along this line, a vision-based ESP anti-cheating solution has been designed and improved for industrial applications.

Challenge 2: Cheating and anti-cheating is an ongoing adversarial game. As cheaters continue to develop new techniques to evade anti-cheating measures, various variants of cheating can render current anti-cheating strategies ineffective and spread in a short period of time with great damage [7]. Anti-cheating systems must be able to recognize and respond to this new cheating in a timely manner. Fully supervised schemes may be insufficient because they cannot detect newly uncovered cheating, and collecting and training new cheating data can be time-consuming after new cheating is manually discovered. Therefore, we propose a framework that unifies both supervised and unsupervised solutions to achieve comprehensive cheating detection.

Challenge 3: Anti-cheating must be subject to a dual audit with human involvement. Even if a powerful model is deployed in practice, incorrect penalties would be unfair and intolerable to normal players. Hence, to ensure fairness and accuracy, the entire anti-cheating procedure must be double-checked with human involvement [13]. To this end, our framework is a human-in-the-loop-based solution that incorporates a dual audit: model initial audit and manual re-audit. In addition, we use model interpretability techniques to provide visual evidence and improve the efficiency of manual involvement.

Our contributions are summarized as follows:

- We propose a vision-based ESP anti-cheating framework for online FPS games, which unifies both supervised and unsupervised solutions to achieve comprehensive cheating detection.
- Based on our proposed framework, we develop a dual-audit human-in-the-loop system for industrial gaming anti-cheating applications.
- Extensive experiments on real-world online FPS game datasets demonstrate the effectiveness and reasonableness of our proposed framework, showing its superiority in both performance and application.

II. Related Work

In this section, we briefly review our work from three aspects: anti-cheating in online FPS games, ESP cheating principle, and visual anomaly detection.

A. Anti-cheating in Online FPS Games

Cheating is widespread in online games [7], [11], [12], particularly in competitive games like online FPS games that are shooting games from a first-person perspective and are often played competitively online. Depending on the benefit gained, common cheating can be divided into three categories [2]–[5]: (1) Aimbots enhance or completely control a player's aim to accurately target the enemy's position. (2) Movement script allows players to abuse game mechanics that increase their movement. (3) Extrasensory perception (ESP), also known as visual hacks (or Wallhacks), provides the player with visual information they would not normally have access to, including enemy positions, health, and equipment. Anti-cheating efforts are based on two main methods: (1) Program-based solutions work much like anti-virus software [6], such as BattlEye[6],

[6]https://www.battleye.com/about/

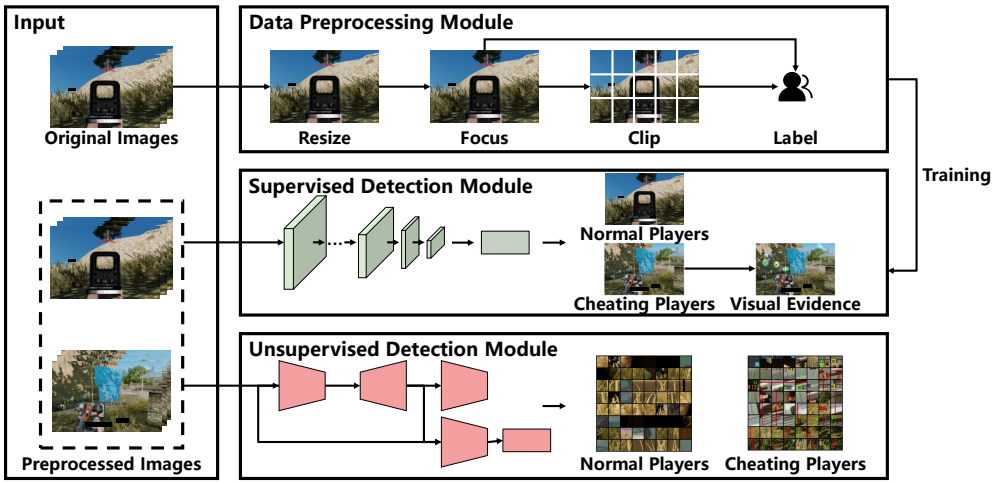


Fig. 3. The overview of our proposed framework VESPA.

Easy Anti-Cheat[7], and Valve Anti-Cheat (VAC) System[8]. They are installed and run on the clients and scan the signature of the game in progress to detect changes to Dynamic-link Library (DLL) files or the effects of known cheats. This type of method is effective in practice but also raises security and privacy concerns. (2) Data-based solutions are designed to classify users as cheating or non-cheating based on various non-visual game data, such as mouse data, keystroke data, behavioral data, and console logs [1], [7]–[10]. Due to data limitations, these methods rely on specific games or scenarios and are insufficient in terms of generality and interpretability.

B. ESP Cheating Principle

ESP is a common form of cheating in online FPS games. External visual elements added to the game interface by heating programs or other sources can assist cheaters in the game and give them a considerable advantage over normal players in the competition. Typically, as shown in Figure 1, the elements include the distance of other players, the box of various types of items, the text of the item description, and so on (this information should normally be inaccessible to the player). In general, online FPS games use a client-server network architecture [2]–[4]. That is, the server is an authoritative host that runs the game to declare the initial game state and rules. The client is the host of each player who connects to the server to play the game. The server maintains the frequency and consistency of game state updates on all clients to provide a better overall game experience. Figure 2 shows the workflow of ESP cheating in online FPS games. Specifically, cheating programs use two main methods to obtain illicit game-related information: ① catching server-to-client packets and ② reading local memory. It then creates a floating window with hints that are added to the original game screen. This way, the cheating information can be visually displayed on the client-side screen as players play the game. Finally, the hint gives the cheating player more extra view to gain a significant competitive advantage over normal players.

C. Visual Anomaly Detection

Anomaly Detection (AD) refers to the problem of finding patterns in given data that do not match the expected patterns [14], which strongly correlates with important computer vision and image processing tasks such as image/video AD [15]. In general, obtaining abnormal samples and detecting novel abnormalities are time-consuming and costly objects in AD. This forces us to develop unsupervised methods for more practical applications. Current unsupervised AD methods are mainly divided into two categories: embedding-based and reconstruction-based methods. The former generally uses a pre-trained model to extract the embedding features of the input image and uses the similarity of the embedding vector between the test image and the reference image as the anomaly score [16]–[19]. Embedding-based methods have recently achieved good performance, but they lack interpretability because it is difficult to clearly distinguish which part of the image causes high abnormal values. In addition, these methods require a high-quality pre-trained model to extract vectors, which is less practical for various industrial applications. The latter employs a generator to reconstruct the input image and calculates the anomaly score based on the reconstruction error [20]–[23]. Typically, Autoencoder (AE) [24] and Generative Adversarial Networks (GAN) [25] are two main ways to reconstruct samples from normal training data. Some recent studies have shown that using adversarial training would improve the generation results [26]–[28]. Despite these efforts, this type of method still suffers from the poor reconstruction ability of the generator, limiting the performance of visual AD.

Remark. In principle, the ESP runs as a process independent of the game client and does not modify the game script files or the memory values of the program. This makes it difficult to detect using traditional program-based methods and data-based methods. In response to the characteristics of the ESP, some game companies identify cheating by obtaining screenshots [2], [11], [12]. Specifically, ESP anti-cheating can

[7]https://www.easy.ac/en-us/
[8]https://help.steampowered.com/en/faqs/view/571A-97DA-70E9-FF74

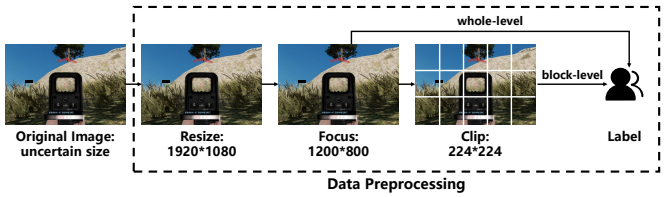


Fig. 4. The four-step preprocessing pipeline in DPM.

be defined as a visual AD problem in terms of whether ESP elements exist in the game interface or not, but with its own peculiarities. Typically, online FPS game interfaces are rich in elements and may contain multiple ESP elements, unlike the AD benchmarks with a single background and limited anomalous elements. In addition, ESP elements are typically local and fine-grained, as opposed to the global anomalies of AD benchmarks in general. For these reasons, the above visual AD methods perform well in AD benchmarks, but may not be applicable to online FPS game data. Inspired by them, our designed and improved vision-based ESP anti-cheating solution is ideal for industrial applications due to its high performance and interpretability.

III. Methodology

In this section, we present the architectural design of our proposed **V**ision-based **ESP A**nti-cheating framework for online FPS games, named VESPA. As illustrated in Figure 3, VESPA consists of three key modules: data preprocessing module, supervised detection module, and unsupervised detection module. We further elaborate on how to unify supervised and unsupervised learning to form a human-in-the-loop system for vision-based ESP anti-cheating in online FPS games.

A. Data Preprocessing Module

The Data Preprocessing Module (DPM) is responsible for acquiring the images from the client and providing data to the subsequent detection modules for training and testing. We have implemented a series of essential privacy measures. Firstly, we adhere to the standard FPS anti-cheating industry protocol[9] for data collection [29] while providing an opt-out option for data collection to all users. Our VESPA system strictly adheres to the Term of Service (ToS) and only captures in-game screenshots. Furthermore, we have applied a three-phase anonymization process that guarantees the irreversibility of the original data [30]. This process encompasses user sampling, data abstraction, and the encryption of Personally Identifiable Information (PII) using salted MD5 (a non-reversing encryption algorithm). Secondly, data security has also been fully integrated into our dual audit procedure. The model initial audit focuses on machines, while the VESPA system utilizes a robust data authorization mechanism to eliminate the risk of data leakage. In the subsequent manual re-audit, the VESPA system pre-processes images to shield privacy information, including masking sensitive parts such as player ID and account information and cropping the image to retain only the crucial parts that require annotation. All personnel accessing data throughout the entire process have signed a Non-Disclosure Agreement (NDA). Thirdly, all data is exclusively employed for anti-cheating and is not used for any other purpose.

Since the original images have different resolutions and noise elements, the image data must be preprocessed as necessary. Figure 4 shows the four-step preprocessing pipeline. (1) ***Resize:*** we resize all the original images from the game client to 1920×1080. (2) ***Focus:*** we adjust the image size to 1200×800 by filtering the irrelevant areas and saving the main part of the image. (3) ***Clip:*** we clip the whole image into $5 \times 3 = 15$ block images with the size of 224×224. (4) ***Label:*** we hire experienced game players or experts to annotate the data. The images containing ESP elements are labeled as positive samples (1) and the others as negative samples (0). In particular, we use data augmentation to generate pseudo ESP cheating samples to increase the number of positive images. Our commonly used techniques include geometric transformations (e.g., cropping, flipping, rotation, and translation) and photometric transformations (e.g., kernel filters, grayscale, and color space translation) [31].

B. Supervised Detection Module

Supervised Detection Module (SDM) aims to automatically learn the differences in visual features between cheating (ESP) samples and normal samples in order to classify them correctly. The challenge is twofold. First, the entire decision-making process is expected to be interpretable. In practice, based on the results of model initial audit, manual re-audit must rely on sufficient credible evidence to finally confirm whether a player is using ESPs and thus penalize him or her. Second, the high cost of acquiring labeled data can lead to data and annotation limitations for supervised learning paradigms. To this end, we introduce three main components to address above these issues, including pre-trained convolutional neural networks, class activation map, and multiple instance learning.

1) Pre-trained Convolutional Neural Networks: Pre-trained Convolutional Neural Network (CNN) has proven its powerful feature extraction capability on visual data and is widely used for various vision tasks in many fields [32]. Typically, Deep Residual Network (ResNet) [33] is a CNN-based neural network with a residual module that can facilitate the training of much deeper networks. Figure 5a shows the ResNet-based binary classifier in SDM. Specifically, we first use the pre-trained ResNet50 as a backbone to obtain the feature map from the input images. Then, we establish the mapping relationship between feature maps and their corresponding categories through a Softmax classifier. Finally, we can obtain the prediction results for each image and visualize the suspected ESP part in the whole original image.

2) Class Activation Map: Class Activation Map (CAM) [34] allows the pre-trained CNN to perform object localization without using any bounding box annotations, which visualizes the predicted class scores for any image and highlights the discriminative object parts detected by the CNN. We define $f_k(x, y)$ as the value of the coordinate (x, y) position in the

[9]https://www.gdcvault.com/play/1026331

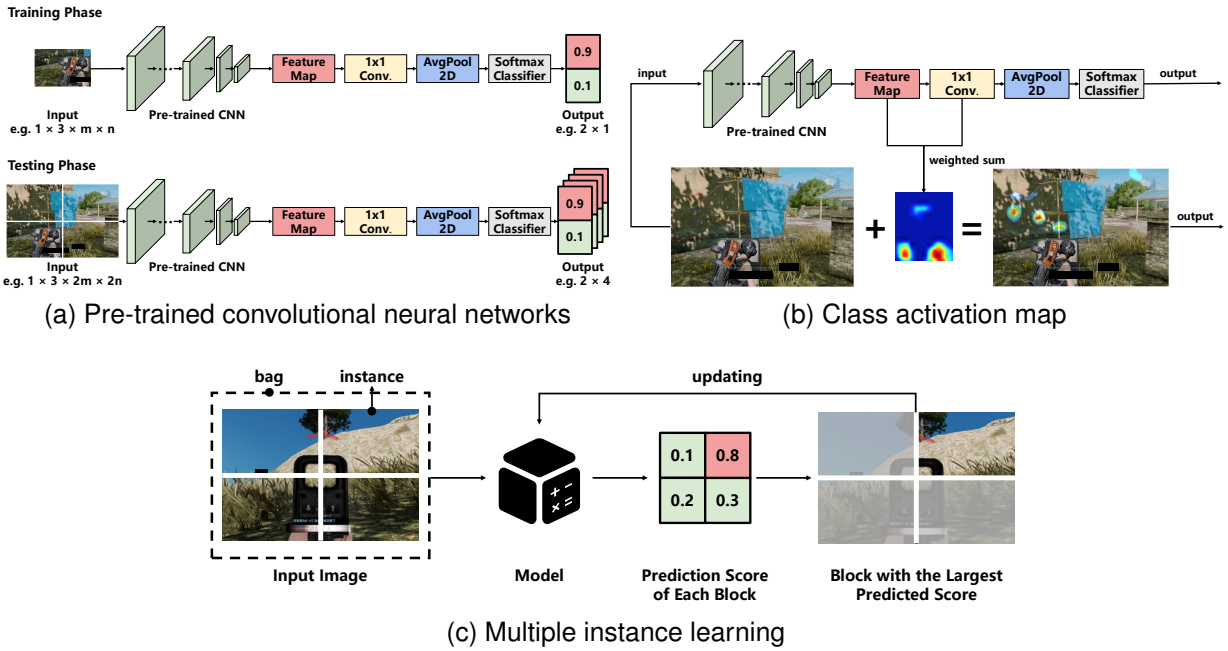


Fig. 5. Three main components in SDM: (a) shows a binary classifier based on pre-trained CNN, (b) shows the procedure of visualizing the ESP elements in a cheating image by CAM, and (c) shows the training procedure of MIL.

k-th output in the last convolutional layer. After the Global Average Pooling (GAP) layer, the output in the k-th feature map is $f_k = \frac{1}{N}\sum_{x,y} f_k(x,y)$, where N is the number of pixels in it. For the category C in the final output, we define $S_c = \sum_k w_k^c F^k$ as the input of the softmax layer, where W_K^C is the weight from the k-th input (F^k) to the final category C. The entire procedure can be described as follows:

$$S_c = \sum_k w_k^c \frac{1}{N}\sum_{x,y} f_k(x,y) = \frac{1}{N}\sum_{x,y}\sum_k w_k^c f_k(x,y). \quad (1)$$

Since S_c indicates the probability that the input image belongs to class C or not, we can get the score of the pixels and decide the importance of the image. Therefore, if these more important feature maps can be visualized, we can find the regions that are more important for the class C. As shown in Figure 5b, CAM can visualize the ESP elements in detail and show their location and position through the heat map to provide interpretability for downstream manual re-audit.

Here we introduce a more efficient inference method, called Fully Convolutional CAM, which allows the use of the original image directly without cropping [29]. Unlike the original CAM method, which employs GAP to obtain the global CAM of the (block) image one by one, the fully convolutional CAM generates a set of block-wise CAMs of the original image simultaneously in a convolutional manner. For this purpose, we replace GAP with 2D average pooling. However, this replacement causes compatibility problems in the downstream linear layers due to size differences between the original image and the block image. To address this issue, we adopt the 1×1 CNN layer to manipulate the channel dimension [35] and use its weights to generate CAMs as shown in Figure 5b [36].

3) Multiple Instance Learning: Multiple Instance Learning (MIL) is a form of weakly supervised learning where training instances are arranged in sets, called bags, and a label is provided for the entire bag [37]. Compared to the traditional supervised learning paradigm where a set of individually labeled instances is input, this method enables the input of a set of labeled bags, each of which contains multiple instances. In this way, the cost of labeled data acquisition can be greatly reduced while maintaining high-quality data utilization. This is important for industrial applications since companies not only have to spend huge costs to label data but also have to pay attention to data privacy, data collection, data computation, and other issues [38]. As shown in Figure 5c, we treat different blocks of the same image as multiple instances belonging to the same bag. For our binary classification tasks, a bag is negative if all instances are labeled as negative, otherwise the bag is positive. Given a bag $X = \{x_1, \ldots, x_n\}$, each $x_i \in X$ is labeled as y_i and the label Y of the bag X is:

$$Y = \begin{cases} +1 & if \quad \exists y_i = +1, \\ -1 & if \quad \forall y_i = -1. \end{cases} \quad (2)$$

On this basis, MIL obtains the prediction score of each instance by the model and allows the model to automatically select the instance with the maximum prediction score to update the parameter in the training phase.

C. Unsupervised Detection Module

Despite achieving impressive performance, the supervisory solution relies heavily on data annotation and therefore has limitations in practical applications. Due to the adversarial game between cheating and anti-cheating, cheating techniques are able to mutate and proliferate into multiple new variants in a very short period of time. In contrast, a supervised solution has difficulty in labeling and training new data in such a short time, which also means that it cannot respond to these uncovered cheating patterns in a timely manner. To this end, the goal of the Unsupervised Detection Module (UDM) is

to automatically discover uncovered ESP elements from a large number of unlabeled images to complement the supervised solution. In practice, UDM achieves multi-granularity hierarchical unsupervised detection through two components, namely block-level detection and whole-level detection.

1) Block-level Detection: Block-level detection primarily uses the Ganomaly [21] algorithm to detect anomalies for blocks cropped from the original images in an unsupervised learning fashion. Ganomaly learns the data distribution for the normal samples through a conditional GAN. As a result, a larger pixel-level distance metric from this learned data distribution at inference time indicates an outlier from that distribution, i.e., an anomaly. However, this method is not directly applicable to our FPS game scenarios for two reasons. First, as shown in Figure 6, cheating elements usually occupy only a small part of the area in cheating blocks, while most of the area is still normal. This means that reconstruction errors in normal areas are very likely to introduce noise. Second, game images are rich in elements and their reconstruction is difficult, so the corresponding endogenous errors can also directly affect the detection results. Hence, we design a novel anomaly detection mechanism. Specifically, we first obtain a differential image $I_{dif} = I_{ori} - I_{gen}$ from the image generated by the generator I_{gen} and the original image I_{ori}. Then, for each of the three channels $C \in \{R, G, B\}$, six gradient features are extracted by the Sobel operator [39] of the differential image I_{dif}, including the first-order mean and variance of the transverse gradients, the first-order mean and variance of the vertical gradients, and the second-order mean and variance of the total gradients:

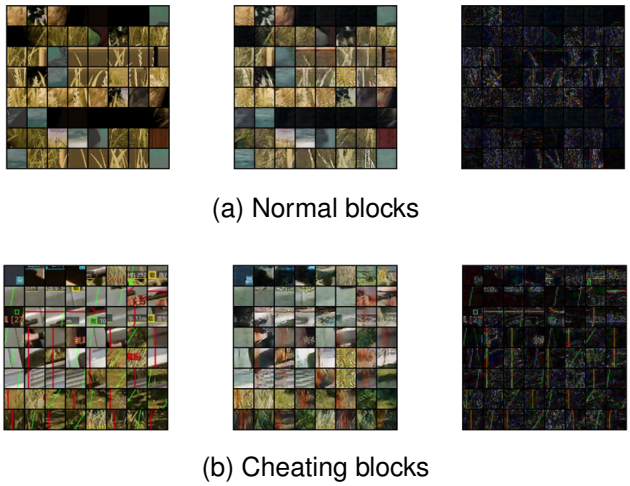
$$\begin{aligned} f^{M,C}_{dx=1} &= Mean(Sobel(I_{dif}, dx=1, dy=0, C)), \\ f^{V,C}_{dx=1} &= Var(Sobel(I_{dif}, dx=1, dy=0, C)), \\ f^{M,C}_{dy=1} &= Mean(Sobel(I_{dif}, dx=0, dy=1, C)), \\ f^{V,C}_{dy=1} &= Var(Sobel(I_{dif}, dx=0, dy=1, C)), \\ f^{M,C}_{dx=dy=2} &= Mean(Sobel(I_{dif}, dx=2, dy=2, C)), \\ f^{V,C}_{dx=dy=2} &= Var(Sobel(I_{dif}, dx=2, dy=2, C)). \end{aligned} \tag{3}$$

This gives us a total of $6 \times 3 = 18$ features for each image, denoted as $f \in F$. Finally, an anomaly score can be derived based on a heuristic approach to determine whether a block I is a cheating block or not. We count the distribution interval $\mathcal{D}_f$ of each feature f in a large number of normal samples. If a feature f is far away from its corresponding distribution intervals $\mathcal{D}_f$, it is considered an anomaly. The overall anomaly score is tracked by weighted voting of all features:

$$score(I) = \sum_{f \in F} w_f \mathbb{I}(f \notin \mathcal{D}_f), \tag{4}$$

where $\mathbb{I}(\cdot)$ is the indicator function and w_f is the weight of feature f. In practice, w_f can be assigned a constant value of 1 or set by game experts.

2) Whole-level Detection: As for the whole image without clipping, we design a whole-level detection based on block-level detection. Specifically, we first detect all blocks of any original image by block-level detection and obtain the anomaly score for each block. Then, we can calculate the total of all block scores as the score of this original image and determine whether the whole image contains ESP elements based on this score. In practical applications, whole-level detection is more straightforward and efficient than block-level detection due to its higher recall (at the expense of lower precision). It is commonly used as a pre-detection tool for coarse-grained detection, as well as to identify and warn about new cheating variants in production. This data-driven approach is advantageous over traditional after-the-fact methods through player reports or other channels (such as social media and game communities) as it allows for timely discovery and response to cheating mutation issues, without having to wait until the cheats have become widespread. On a daily basis, the TOP-N images with the highest whole-level detection anomaly scores are output and manually confirmed by game experts to identify new cheating variants that are different from the past. They are then submitted to game operations for further processing.



(a) Normal blocks

(b) Cheating blocks

Fig. 6. The samples of the raw image, the generated image, and the differential image for block-level detection.

D. Human-in-the-loop System

Our work has been implemented and deployed in production for an extended period, serving the main traffic in many online FPS games released from NetEase Games[10]. Figure 7 illustrates the human-in-the-loop workflow of our proposed VESPA system. Due to the peculiarities of the business in practice, we are very cautious about the cheating detection results, which requires us to not fully rely on models to automate penalties for cheaters. Any incorrect penalty is unfair and intolerable to normal players and can seriously damage the reputation of the game company. Therefore, the entire anti-cheating procedure must be double-checked with human involvement before cheaters can be penalized.

To this end, our system is a human-in-the-loop-based solution that incorporates a dual audit: model initial audit and manual re-audit. The system first receives the aggregated cheating detection results through supervised and unsupervised solutions. Based on this basis, the human then confirms the

[10]https://www.neteasegames.com

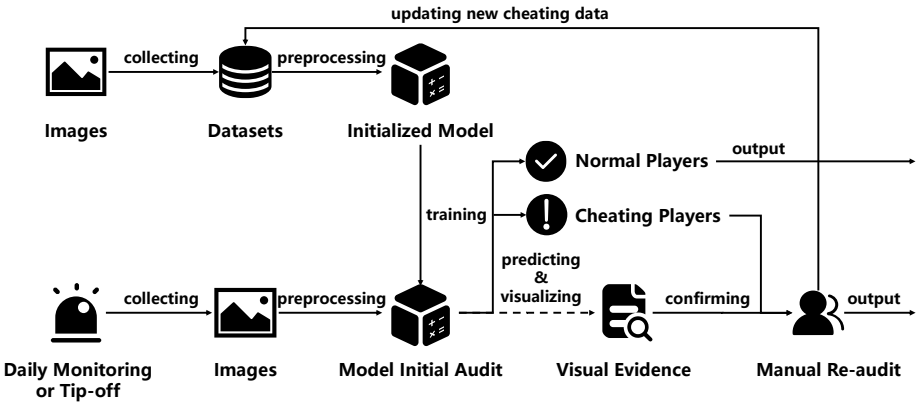


Fig. 7. The overview of our human-in-the-loop system.

TABLE I
DETAILED DESCRIPTION OF THE ONLINE FPS GAME DATASET, WHERE DA DENOTES DATA AUGMENTATION.

	Size	#Samples	#Neg.	#Pos.	DA
Train	224×224	81,928	40,964	40,964	✓
Valid	224×224	20,000	10,000	10,000	×
Test	1200×800	20,000	10,000	10,000	×

TABLE II
PERFORMANCE COMPARISON IN SUPERVISED AND BLOCK-LEVEL UNSUPERVISED DETECTION.

Methods	Precision↑	Recall↑	F1-score↑
Expert Model-interface	0.9771	0.3929	0.5604
Expert Model-geometry	0.9236	0.1003	0.1810
Expert Model-HP	0.9767	0.4248	0.5921
Expert Model-all	0.9242	0.8558	0.8887
VGG	0.7831	0.8758	0.8269
DenseNet	**0.9942**	0.7866	0.8783
Wallhack Detector	0.9028	0.7997	0.8481
VESPA-SDM	0.9131	**0.9165**	**0.9148**
VESPA-SDM$_{part}$	0.8371	0.7526	0.7926
VESPA-SDM$_{full}$	0.7846	0.6912	0.7349
Ganomaly	0.7101	**0.7580**	0.7332
VESPA-UDM$_{block}$	**0.8100**	0.7300	**0.7700**

results one by one. Note that in the past, manual re-audit required tracking down other information related to the cheater's account to complete the confirmation, which could take a long time. Now, we can effectively improve the efficiency of this process by providing visual evidence using the model interpretability techniques in Section III-B2. In this process, the uncovered cheating data in the detection results of the unsupervised solutions are also manually confirmed as new data to train the supervised solutions. In this way, the whole system is able to iterate itself from a cycle of discovering new cheating, learning cheating data, and detecting cheating.

IV. EXPERIMENTS

In this section, we conduct comprehensive experiments on a real-world online FPS game dataset. First, we describe the experimental setup. Then, we evaluate VESPA in terms of both supervised and unsupervised detection. Finally, we present the system evaluation of VESPA deployed in production.

A. Experimental Setup

1) Datasets: Our real-world anonymous data is collected from an online FPS game provided by NetEase Games, under the premise of ensuring data privacy. After necessary preprocessing, the dataset contains 121,928 images of 60,964 cheating images with ESP elements and 60,964 normal images. Table I summarizes the basic dataset statistics.

2) Baselines: For supervised detection, we select several typical groups of state-of-the-art (SOTA) baselines:

- **Expert Models.** Game experts design visual features for rule detection based on the characteristics of various ESP elements, including interface (cheating software interface or operation panel), geometry (displaying enemy positions through geometric shapes such as rays, boxes, circles, and skeletons), and HP (enemy health).
- **Computer Vision (CV) Models.** Based on the pre-training and fine-tuning paradigm, supervised learning of binary classification tasks is performed on widely used pre-training models, such as VGG16 and DenseNet121.
- **Wallhack Detector [29].** It is currently the most advanced method, which trains a binary classification model using ResNet50 and CAM to detect Wallhack cheating.
- **Variants.** We construct the following variants to investigate the effect of different labeled data settings on the supervised detection performance: (1) VESPA-SDM follows the same SDM settings proposed in VESPA, where the whole process uses full training data for training in a MIL fashion. (2) VESPA-SDM$_{part}$: First, we randomly split the training set into two parts. The model is then trained in a MIL fashion using the small subset with annotations and the remaining large subset without annotations. In our experiments, the small subset contains 6,000 positive images and 6,000 negative images and the remaining 69,928 images are used as the large subset. (3) VESPA-SDM$_{full}$: Similar to VESPA-SDM$_{part}$, we also train the MIL-based model on two subsets of the training set. The difference is that instead of using real annotations of the small subset, we use pseudo-annotations generated by clustering by features extracted from pre-trained CNN models. Thus, this is a (pseudo-) unlabeled scheme as no data annotation is used in both subsets.

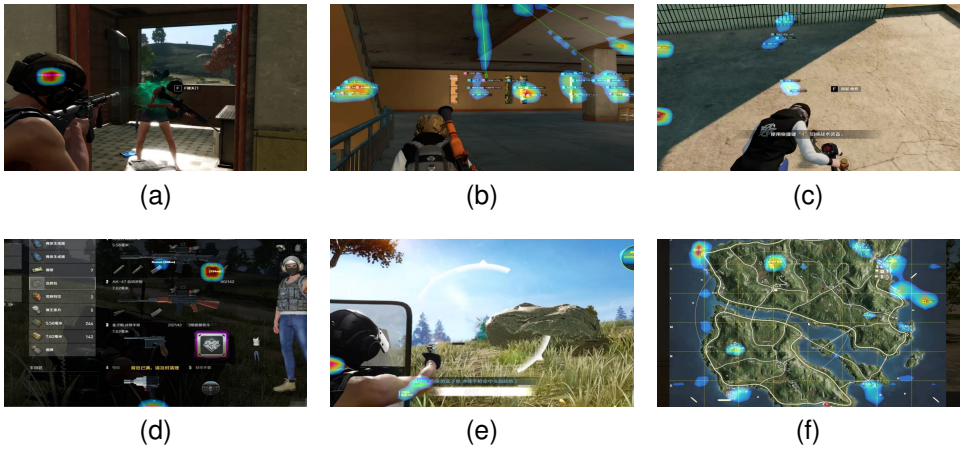


Fig. 8. The visualization results of VESPA-SDM.

For unsupervised detection, we mainly compare our method to Ganomaly [21]. Ganomaly learns the data distribution for the normal samples through a conditional GAN. A larger pixel-level distance metric from this learned data distribution at inference time indicates an anomaly.

3) Evaluation Metrics: We evaluate VESPA from two aspects, namely effectiveness and efficiency. In terms of effectiveness, we formalize both supervised and unsupervised detection as binary classification tasks and use three common metrics for evaluation, including precision, recall, and F1-score. In particular, we apply AUC (the area under the curve) to evaluate the classification performance for whole-level unsupervised detection. In production, this method is mainly employed as a pre-detection tool or for alerting new cheating variants, so we only focus on the ranking performance of the model. As for efficiency, we focus on the computational time for the whole detection procedure in the online deployment system, which mainly consists of two phases: model initial audit and manual re-audit. First, we compare the model initial audit time on CPU and GPU platforms for both supervised and unsupervised detection, respectively. Then, we compute the manual re-audit time and compare the impact of visual evidence on its efficiency. We use the percentage of the difference between the two as an evaluation metric:

$$R_t = \frac{\Delta t}{t_{w/o}}, \quad \Delta t = t_{w/o} - t_{w/}. \tag{5}$$

Here $t_{w/o}$ and $t_{w/}$ are the average time to manually audit an image with or without visual evidence, respectively.

B. Supervised Detection Evaluation

We perform quantitative and qualitative evaluations to verify the superiority of VESPA in supervised detection.

1) Quantitative Evaluation: Table II shows the experimental results of supervised detection. Overall, VESPA-SDM performs best among all baselines because it utilizes all labeled data for the training. Expert models are labor and experience intensive. These methods have high precision and low recall, especially a lack of sufficient generalization ability to new cheating variants. CV models can automatically learn effective features from massive data without external feature engineering, and generalize well. Wallhack Detector can provide interpretability while maintaining high performance, which is very critical in practical applications. There is also a large amount of unlabeled data in production. Here, VESPA-SDM can also perform well. The experimental results show that VESPA-SDM$_{part}$ and VESPA-SDM$_{full}$ also achieve relatively good performance with different labeled data settings, respectively. In conclusion, VESPA-SDM can be applied to supervised ESP anti-cheating in online FPS games.

2) Qualitative Evaluation: To further examine the performance in supervised detection, we visualize the results of VESPA-SDM in Figure 8 and discuss the following questions:

- ***Q1: Does VESPA-SDM perform satisfactorily in detecting cheating?*** We evaluate the percentage of ESP elements detected by VESPA-SDM in the cheating images. The majority of ESP elements in each cheating image can be detected correctly. For example, there is only one ESP element in Figure 8a and VESPA-SDM found its location correctly. There are 22 ESP elements in Figure 8b and 100% of them are detected by VESPA-SDM.
- ***Q2: Can VESPA-SDM distinguish between normal and ESP elements?*** We randomly select two cheating images that contain normal elements similar to the ESP elements, such as the text "pick up" or "go straight" regarding game instructions. For example, there is a large amount of text information in Figure 8c and Figure 8d. Most of the text is normal elements, such as weapon names, props, and so on. The experimental result shows that no normal elements are displayed by mistake. This indicates that VESPA-SDM can effectively distinguish between ESP elements and normal text information, even when they are visually very similar.
- ***Q3: Does VESPA-SDM have a strong generalization capability?*** Uncovered but similar cheating can still be detected and visualized by VESPA-SDM. For example, the element in the upper right of Figure 8e never appears in the training set. However, some similar ESP elements exist in the training set and have been learned by VESPA-SDM. In this case, the VESPA-SDM is able to generically detect the uncovered ESP elements.

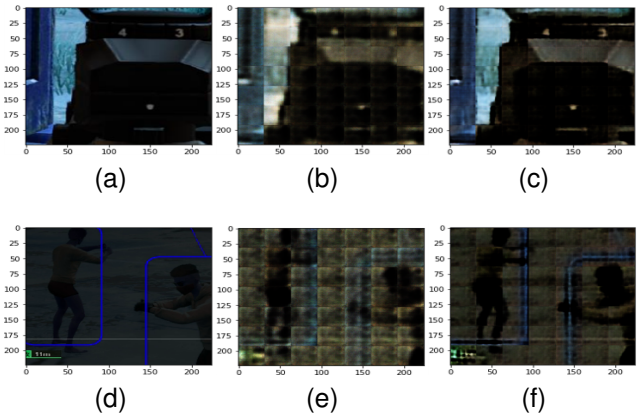


Fig. 9. The visualization results of VESPA-UDM.

TABLE III
RESULTS OF THE WHOLE-LEVEL UNSUPERVISED DETECTION.

Method	AUC↑
Ganomaly	0.6320
VESPA-UDM$_{whole}$	**0.6790**

- ***Q4: How to deal with the bad case?*** We also find some bad cases from the detection results of VESPA-SDM. Figure 8f is a screenshot of a map in online FPS games. VESPA-SDM correctly visualizes the ESP elements, but also incorrectly visualizes many normal elements, such as UI elements, entity names, coordinates, and appearances. This may be due to the fact that these normal elements are less available in our training set such that the model cannot fully learn their features. Therefore, more diverse correctly annotated images need to be used to improve the generalization of the model. By the way, such bad cases are corrected in the manual re-audit to avoid incorrect detection and iterative supervised solutions.

C. Unsupervised Detection Evaluation

Similarly, we evaluate the unsupervised detection performance of VESPA in both quantitative and qualitative terms.

1) Quantitative Evaluation: We perform experiments on block-level and whole-level detection, respectively. First, Table II shows that VESPA-UDM$_{block}$ achieves a precision of 0.8100, a recall of 0.7300, and an F1-score of 0.7700 in block-level unsupervised detection, which is a great improvement over Ganomaly. Compared to the VESPA-SDM, VESPA-UDM$_{block}$ significantly outperforms the (pseudo-) unlabeled scheme VESPA-SDM$_{full}$ and achieves competitive performance with the weakly labeled scheme VESPA-SDM$_{part}$. Second, Table III shows that VESPA-UDM$_{whole}$ achieves an AUC of 0.6790 in whole-level unsupervised detection, which is significantly higher than Ganomaly. As a coarse-grained pre-detection solution in practice, these results have met the expected performance requirements.

2) Qualitative Evaluation: We present the process of image generation by VESPA-UDM$_{block}$ and further discuss the difference between visual AD and ESP anti-cheating in Figure 9.

TABLE IV
COMPUTATIONAL PERFORMANCE IN CPU AND GPU PLATFORM.

Method	CPU	GPU
VESPA-SDM	2.000s	0.066s
VESPA-UDM	1.240s	0.070s

TABLE V
THE ACCELERATION EFFECT ON MANUAL RE-AUDIT.

$t_{w/o}$	$t_{w/}$	Difference	R_t
2.31s	1.24s	1.07s	0.4632

- ***Q1: Does VESPA-UDM perform satisfactorily in detecting cheating?*** Throughout the whole process of image generation, the ESP elements remain distinctly different from other elements in the image in terms of generative power, which provides the basis for unsupervised detection. For example, Figure 9a and Figure 9d are two raw cheating images containing ESP elements. After the 1 epoch, the generated images of ESP elements are blurred, as shown in Figure 9b and Figure 9e. After the 5 epoch, we can see that the ESP elements are gradually generated clearly in Figure 9c and Figure 9f.
- ***Q2: Why not adopt the Ganomaly for ESP anti-cheating directly?*** Unlike general visual AD tasks, ESP anti-cheating focuses more on local differences. In contrast, Ganomaly computes the global differences between generated images and raw images to detect anomalies, ignoring many local details. As a result, ESP elements such as lines, boxes, and small circles are difficult to detect. To this end, we design a novel mechanism for ESP anti-cheating based on Ganomaly, refer to Section III-C1.

D. System Evaluation

From a system perspective, we focus on the scalability and efficiency of VESPA in two phases of practical applications: model initial audit and manual re-audit. First, we measure the runtime of the VESPA-SDM and the VESPA-UDM on CPU and GPU platforms, respectively. The experimental results are shown in Table IV. VESPA-SDM completes within 0.066 seconds and VESPA-UDM completes within 0.070 seconds in the GPU environment, significantly outperforming the CPU environment. This shows that VESPA is efficient in practice and can be applied to large-scale ESP anti-cheating platforms. Second, Table V shows the acceleration effect in the manual re-audit process by visual evidence obtained from CAM. In general, the average time for auditing a cheating image is 2.31 seconds, while it is only 1.24 seconds with the help of VESPA. This is because the CAM proposed in VESPA can visualize the location of ESP elements by heat map to provide reliable visual evidence. In this way, auditors can perform re-audit work more easily and efficiently, with an average reduction of 1.07 seconds per image and an acceleration of 46.32%.

V. CONCLUSION

This paper investigates the ESP anti-cheating problem in online FPS games. Specifically, we propose a vision-based

anti-cheating framework that unifies both supervised and unsupervised solutions to achieve comprehensive cheating detection. We then design and implement a dual-audit human-in-the-loop system for industrial applications. We demonstrate the superiority of our proposed framework in both performance and applicability by evaluating it online and offline.

ACKNOWLEDGMENTS

This work is supported by the Key Research and Development Program of Zhejiang Province (No. 2022C01011).

REFERENCES

[1] H. Alayed, F. Frangoudes, and C. Neuman, "Behavioral-based cheating detection in online first person shooters using machine learning techniques," in *2013 IEEE conference on computational inteligence in games (CIG)*. IEEE, 2013, pp. 1–8.

[2] A. Jonnalagadda, I. Frosio, S. Schneider, M. McGuire, and J. Kim, "Robust vision-based cheat detection in competitive gaming," *Proceedings of the ACM on Computer Graphics and Interactive Techniques*, vol. 4, no. 1, pp. 1–18, 2021.

[3] S. Khalifa, "Machine learning and anti-cheating in fps games," *Master's thesis*, 2016.

[4] R. Spijkerman and E. Marie Ehlers, "Cheat detection in a multiplayer first-person shooter using artificial intelligence tools," in *2020 The 3rd International Conference on Computational Intelligence and Intelligent Systems*, 2020, pp. 87–92.

[5] M. Willman, "Machine learning to identify cheaters in online games," 2020.

[6] A. Philbert, "Dectecting cheating in computer games using data mining methods," *American Journal of Computer Science and Information Technology*, 2018.

[7] J. Tao, J. Xu, L. Gong, Y. Li, C. Fan, and Z. Zhao, "Nguard: A game bot detection framework for netease mmorpgs," in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 2018, pp. 811–820.

[8] S. Zhao, J. Fang, S. Zhao, R. Wu, J. Tao, S. Li, and G. Pan, "T-detector: A trajectory based pre-trained model for game bot detection in mmorpgs," in *2022 IEEE 38th International Conference on Data Engineering (ICDE)*. IEEE, 2022, pp. 992–1003.

[9] Q. Zhang, "Improvement of online game anti-cheat system based on deep learning," in *2021 2nd International Conference on Information Science and Education (ICISE-IE)*. IEEE, 2021, pp. 652–655.

[10] J. P. Pinto, A. Pimenta, and P. Novais, "Deep learning and multivariate time series for cheat detection in video games," *Machine Learning*, vol. 110, no. 11-12, pp. 3037–3057, 2021.

[11] J. Tao, Y. Xiong, S. Zhao, Y. Xu, J. Lin, R. Wu, and C. Fan, "Xai-driven explainable multi-view game cheating detection," in *2020 IEEE Conference on Games (CoG)*. IEEE, 2020, pp. 144–151.

[12] J. Tao, Y. Xiong, S. Zhao, R. Wu, X. Shen, T. Lyu, C. Fan, Z. Hu, S. Zhao, and G. Pan, "Explainable ai for cheating detection and churn prediction in online games," *IEEE Transactions on Games*, 2022.

[13] F. M. Zanzotto, "Human-in-the-loop artificial intelligence," *Journal of Artificial Intelligence Research*, vol. 64, pp. 243–252, 2019.

[14] G. PANG, C. SHEN, L. CAO, and A. V. D. HENGEL, "Deep learning for anomaly detection: A review," *ACM Computing Surveys*, vol. 54, no. 2, p. 1, 2021.

[15] B. Mohammadi, M. Fathy, and M. Sabokrou, "Image/video deep anomaly detection: A survey," *arXiv preprint arXiv:2103.01739*, 2021.

[16] J. Andrews, T. Tanay, E. J. Morton, and L. D. Griffin, "Transfer representation-learning for anomaly detection." JMLR, 2016.

[17] T. Defard, A. Setkov, A. Loesch, and R. Audigier, "Padim: a patch distribution modeling framework for anomaly detection and localization," in *Pattern Recognition. ICPR International Workshops and Challenges: Virtual Event, January 10–15, 2021, Proceedings, Part IV*. Springer, 2021, pp. 475–489.

[18] J. Yi and S. Yoon, "Patch svdd: Patch-level svdd for anomaly detection and segmentation," in *Proceedings of the Asian Conference on Computer Vision*, 2020.

[19] K. Roth, L. Pemula, J. Zepeda, B. Schölkopf, T. Brox, and P. Gehler, "Towards total recall in industrial anomaly detection," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 14 318–14 328.

[20] T. Schlegl, P. Seeböck, S. M. Waldstein, U. Schmidt-Erfurth, and G. Langs, "Unsupervised anomaly detection with generative adversarial networks to guide marker discovery," in *International Conference on Information Processing in Medical Imaging*. Springer, 2017, pp. 146–157.

[21] S. Akcay, A. Atapour-Abarghouei, and T. P. Breckon, "Ganomaly: Semi-supervised anomaly detection via adversarial training," in *Asian Conference on Computer Vision*. Springer, 2018, pp. 622–637.

[22] P. Perera, R. Nallapati, and B. Xiang, "Ocgan: One-class novelty detection using gans with constrained latent representations," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 2898–2906.

[23] Y. Liang, J. Zhang, S. Zhao, R. Wu, Y. Liu, and S. Pan, "Omni-frequency channel-selection representations for unsupervised anomaly detection," *IEEE Transactions on Image Processing*, vol. 32, pp. 4327–4340, 2023.

[24] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," in *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*, Y. Bengio and Y. LeCun, Eds., 2014.

[25] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in Neural Information Processing Systems*, Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K. Weinberger, Eds., vol. 27. Curran Associates, Inc., 2014.

[26] ——, "Generative adversarial nets," in *Advances in neural information processing systems*, 2014, pp. 2672–2680.

[27] D. Pathak, P. Krahenbuhl, J. Donahue, T. Darrell, and A. A. Efros, "Context encoders: Feature learning by inpainting," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 2536–2544.

[28] Y. Liang, J. Zhang, S. Zhao, R. Wu, Y. Liu, and S. Pan, "Omni-frequency channel-selection representations for unsupervised anomaly detection," *IEEE Transactions on Image Processing*, vol. 32, pp. 4327–4340, 2023.

[29] J. Hwang, "Beating wallhacks using deep learning with limited resources," *Game Developers Conference*, 2019, https://www.gdcvault.com/play/1026331/ML-Tutorial-Day-Beating-Wallhacks.

[30] Y. Zhu, D. Xi, B. Song, F. Zhuang, S. Chen, X. Gu, and Q. He, "Modeling users' behavior sequences with hierarchical explainable network for cross-domain fraud detection," in *Proceedings of The Web Conference 2020*, 2020, pp. 928–938.

[31] C. Shorten and T. M. Khoshgoftaar, "A survey on image data augmentation for deep learning," *Journal of big data*, vol. 6, no. 1, pp. 1–48, 2019.

[32] Z. Li, F. Liu, W. Yang, S. Peng, and J. Zhou, "A survey of convolutional neural networks: analysis, applications, and prospects," *IEEE transactions on neural networks and learning systems*, 2021.

[33] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[34] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba, "Learning deep features for discriminative localization," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 2921–2929.

[35] M. Lin, Q. Chen, and S. Yan, "Network in network," in *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*, Y. Bengio and Y. LeCun, Eds., 2014.

[36] X. Zhang, Y. Wei, J. Feng, Y. Yang, and T. S. Huang, "Adversarial complementary learning for weakly supervised object localization," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 1325–1334.

[37] M.-A. Carbonneau, V. Cheplygina, E. Granger, and G. Gagnon, "Multiple instance learning: A survey of problem characteristics and applications," *Pattern Recognition*, vol. 77, pp. 329–353, 2018.

[38] T. Fredriksson, D. I. Mattos, J. Bosch, and H. H. Olsson, "Data labeling: An empirical investigation into industrial challenges and mitigation strategies," in *Product-Focused Software Process Improvement: 21st International Conference, PROFES 2020, Turin, Italy, November 25–27, 2020, Proceedings 21*. Springer, 2020, pp. 202–216.

[39] R. O. Duda, P. E. Hart *et al.*, *Pattern classification and scene analysis*. Wiley New York, 1973, vol. 3.