Abnormal Data Detection Model Based on Autoencoder and Random Forest Algorithm: Camera Sensor Data in Autonomous Driving Systems

Geng Shengwen, Mohd Hafeez Osman*
Faculty of Computer Science and Information Technology, Universiti Putra Malaysia, 43400 Serdang, Malaysia

***Abstract*—This project develops an AI-based anomaly detection system. In the field of autonomous driving, abnormal data will directly affect the safety of autonomous driving systems, especially in terms of abnormal camera sensor data. Sensor failure, environmental changes, or bad weather can lead to the emergence of abnormal data, which can affect the decision-making process and may have disastrous consequences. Based on the above problems, this study addresses this challenge by proposing a hybrid anomaly detection model (called CAE-RF) that combines convolutional autoencoders and random forest algorithms to achieve efficient and accurate identification of abnormal data patterns to improve the safety of autonomous driving systems. The proposed method will use convolutional autoencoders to calculate the reconstruction error and combine the hidden features extracted by the encoder as the input of the random forest to distinguish normal data from abnormal data. The key performance indicators such as accuracy, precision, recall, and F1 score are used to evaluate the model, and the robustness is guaranteed by cross-validation. Experimental results show that the CAE-RF model has an accuracy of 92% in distinguishing normal and abnormal data. Compared with traditional methods, the CAE-RF model achieves higher accuracy and reliability. The implementation of this model can timely identify and process abnormal data, reduce the risks brought by sensor failure or external environment changes, prevent potential accidents, and improve the safety and reliability of the autonomous driving system.**

***Keywords*—Automatic driving; anomaly data detection; convolutional autoencoder; random forest; CAE-RF**

I. Introduction

A. Project Overview

With the rapid development and application of automatic driving technology, the safety of automatic driving has become the focus of attention. The reason why self-driving cars have not been widely used lies in their safety problems [1-3]. Therefore, how to ensure the safety of self-driving cars is an important research topic. One of the key factors affecting the safety of automatic driving is data security, and the correctness and accuracy of data will directly affect the safety of automatic driving system, thus affecting the safety of vehicles and passengers. However, due to various factors such as sensor failure, environmental anomalies, weather conditions, etc. [4], the occurrence of anomaly data is inevitable. An anomaly here is defined as an observation that deviates substantially from some established notion of normal. [5] Therefore, we need an anomaly data detection model to find anomaly data in time. Anomaly detection model can identify abnormal patterns in massive data mining, so it can well detect and respond to potential sensor faults, ensure the normal operation and safety of the system, and avoid accidents. This study will focus on camera sensor data, namely image data. Image data anomalies mainly include noise, overexposure, low brightness, occlusion and other anomaly types. Machine learning algorithms can learn more complex patterns and are able to spot anomalies hidden in the data. And machine learning algorithm can automatically learn the patterns and features in the data, so as to detect anomalies quickly and accurately. Compared with the traditional anomaly detection technology, this greatly improves the efficiency and accuracy of detection. Therefore, this study aims to develop an efficient and accurate model for anomaly detection in image data using machine learning techniques.

*Corresponding Author

This paper is organized as follows: Section II reviews related studies; Section III introduces the proposed method; Section IV presents the experimental procedure; Section V analyzes the results; and Section VI discusses the paper.

B. Problem Statements

Autonomous vehicles have emerged as a promising future transportation technology. However, ensuring their safety and security remains a major challenge. Anomalous data is one of the major issues that could threaten the normal operation of driverless vehicles, which could lead to sensor data errors that lead to faulty navigation decisions, resulting in accidents and deaths [4].

Traditional techniques heavily rely on a strong understanding of the "ground truth" to establish a clear and measurable definition of anomalies. However, in many real-world scenarios where data models change frequently over time, these techniques often fail to deliver satisfactory performance despite their complexity [6].

Detection models under supervised learning are not reliable for unexpected or rare anomalies that do not occur during training. Because unsupervised learning lacks an annotated model to explicitly distinguish between normal and abnormal data, detection models experience a higher proportion of false positives and false negatives [4].

C. Project Objectives

The objective of this research is to develop a model for detecting anomalies in camera sensor data in an autonomous driving system, which will extract features from the collected data and then identify the anomalies based on the features of the anomalous data, and give alerts after identifying the anomalies. The main objectives of the project are as follows:

PO1: The camera sensor data is collected, key features are extracted, and anomalous data is identified based on these features.

PO2: Use machine learning technology to improve the accuracy and efficiency of anomaly data detection to cope with changing environments.

PO3: The abnormal data monitoring model is developed by combining autoencoder and random forest to reduce the disadvantages of unsupervised learning and supervised learning, enlarge their advantages and improve the reliability of the model.

D. Scope of the Project

- Research and analyze the current application of anomaly data detection technology.
- Collecting Camera Sensor Data in Autonomous Driving Systems.
- Construction, training and verification of anomaly data detection model.
- Evaluate the accuracy and effectiveness of the anomaly data detection model.

II. Related Work

A. Conventional Anomaly Data Detection Techniques

There are two types of statistical methods: parametric and non-parametric. Parametric statistical methods estimate the parameters based on the data and presume that the underlying distribution of the data is known, such as Gaussian models, regression models, or mixed parametric distribution methods [7]. Nonparametric statistical techniques do not assume a known distribution, but they determine the distribution based on the data itself, such as methods based on histograms and kernel functions [8]. A thorough analysis of the various statistical methods used for novelty identification can be found in the research in study [9]. It encompasses non-parametric techniques like k-NN based, Parzen density estimation, string matching, and clustering as well as parametric techniques like hidden Markov models, hypothesis testing, and probabilistic and Gaussian mixture modeling. Furthermore, statistical methods are not very generic when dealing with high-dimensional data, despite their advantage in being interpretable and explicable. In the case of high-dimensional data, machine learning techniques can do better than statistical techniques.

Second, Data in sensor systems are typically generated in the form of time series, and time series analysis (TSA) is used to extract statistical features and make predictions about future values. Anomalies can be detected by comparing the difference between the actual and predicted values. Commonly used methods include cross-correlation analysis, autoregressive moving average (ARMA), autoregressive integral moving average (ARIMA), Kalman filtering, etc. [10].

Although time series analysis is simple and effective in dealing with additive outliers, it is less effective in detecting anomalies caused by "drastic" changes and is mainly suitable for "moderate" anomaly events.

B. ML for Anomaly Detection

The study in [8] proposed three basic methods to solve the problem of outlier detection, namely:

a) Monitoring: Modeling normal and anomaly; It requires labeled data for each category.

b) Unsupervised: Anomalies are identified without prior knowledge of the data.

c) Semi-supervised: only normality is modeled; Determine anomalies based on their departure from the typical threshold; another name for it is novelty recognition or detection.

At present, the commonly used supervised learning algorithms mainly include proximity-based classifiers [11], support vector machines (SVM) [12], decision trees [13-14], Random forests [15], and rule-based classifiers [16].

Surveillance techniques demonstrate strong robustness due to their reliance on pre-labeled data as the "ground truth." However, in many real-world systems, such data is either limited or entirely unavailable. To address this challenge, semi-supervised and unsupervised methods have been introduced, effectively bridging the gap.

The underlying premise of unsupervised learning algorithms is that outliers are uncommon and substantially distinct from typical occurrences [17]. Cluster-based approaches, which employ similarity metrics to group data instances, are among the most often used techniques. A data instance is considered an exception if it is not a part of a cluster or if its cluster is much smaller than another cluster. In [18], the authors propose a global outlier detection technique that uses clustering to detect sensor node anomalies.

The autoencoder is another widely used unsupervised learning algorithm [19]. It is trained exclusively on normal data, enabling the model to reconstruct inputs with minimal reconstruction error. During the detection phase, anomalies are identified as instances with higher reconstruction errors, as the model has not encountered these patterns during training. Thresholds are defined to capture and classify these anomalous data points.

A semi-supervised learning algorithm, Single-class SVM (OC-SVM) is a semi-supervised SVM that does not require exception labels. It is applied in study [20] to attack detection in sensor networks in smart cities.

Although semi-supervised learning is optimal when very little labeled data is available, the assumptions associated with using unlabeled data create some limitations. Inaccurate assumptions may result in subpar performance because they rely on the link between labeled and unlabeled data distributions.

Through literature review, we have a basic understanding of the basic working principle of autonomous vehicles, and analyze the safety and reliability of autonomous driving systems. Then, the conventional anomaly detection technology and the anomaly detection technology applying machine learning technology are studied. Through the comparison between machine learning technology and conventional traditional anomaly detection technology, it is found that in many time scenarios where the data model changes greatly over time, the conventional anomaly detection technology cannot bring satisfactory performance. In order to accommodate the dynamic of the big data paradigm, this necessitates the incorporation of machine learning techniques at the tradeoff of less strict formalization [6]. Therefore, this paper decided to use a combination of autoencoder (unsupervised learning approach) and random forest (supervised learning approach) techniques to develop anomaly data detection models.

III. Methodology

This chapter is divided into three sections, each of which will describe the specific tasks of each phase. These include data preprocessing, model development, model validation and evaluation, and documentation. Fig. 1 shows the three phases of the project process.

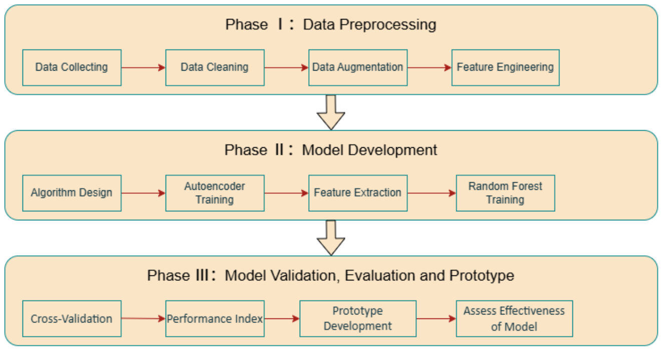

Fig. 1. The framework of the research.

A. Data Preprocessing

Data preprocessing is a key step in ensuring the quality and reliability of the datasets used for model training and testing.

It involves several stages, including data collection, cleaning, integration and standardization. The goal is to transform the raw data into a structured and meaningful format suitable for machine learning algorithms.

The first step was data collection, where camera sensor data were collected from the A2D2 public datasets. These datasets provide a variety of scenarios, such as different weather and lighting conditions.

1) Data cleaning process: The first step is to remove invalid samples. During the initial inspection, it was found that some image files may be damaged (such as unable to load) or the format does not meet the requirements (such as grayscale images instead of RGB images). Through automated script detection, all image files that cannot be loaded normally or have incorrect formats are removed. The second step is to deal with duplicate images. The dataset may contain duplicate images, which will cause overfitting or classification bias in the model during training. By calculating the hash value of each image, completely duplicate images are detected and removed.

Then comes the data enhancement phase, in machine learning and deep learning tasks, especially in image classification and anomaly detection, the quality and quantity of data play a crucial role in the performance of the model. However, we often face the problem of insufficient data or a single data distribution in practical applications, especially in the anomaly detection task, where the anomaly data itself is extremely scarce and the normal data may have an insufficient number of samples or an incomplete coverage of the feature space at the time of collection. Therefore, data transformation and data enhancement techniques are used in the study to generate anomaly data. The following anomaly types are included:

a) Noise: Add random Gaussian noise to the image, with the noise intensity taking a random value with a standard deviation of 0.01 to 0.05 to simulate the interference during sensor acquisition.

b) Rotation: Randomly rotate the image clockwise or counterclockwise by 90° to 180°. This operation can simulate the rotation phenomenon of abnormal objects caused by changes in camera angle during image acquisition.

c) Color_shift: Randomly perturb the hue, saturation, and contrast of the image to enhance the robustness of the model to color change anomalies.

d) Brightness: Randomly increase or decrease the image brightness, ranging from 80% to 120% of the original brightness, to simulate anomaly detection scenarios under different lighting conditions.

e) Occlusion: Randomly add irregular occlusion areas to the image, with the occlusion area accounting for 30% to 80% of the total image area, to simulate abnormal patterns caused by perspective occlusion or obstacle occlusion.

f) Blur: Applies a Gaussian blur with a blur radius of 1 to 3 pixels, simulating a blurred image caused by out-of-focus effects. The examples of images is shown in Fig. 2.

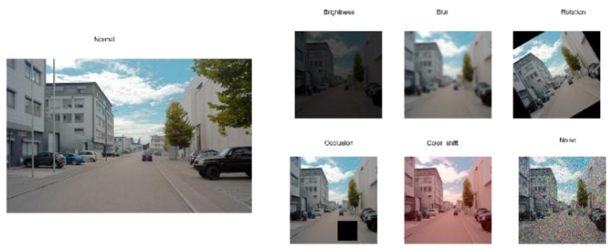

Fig. 2. Example of Images.

Finally, there is the feature engineering phase, where the size and format of the images in the dataset are usually inconsistent due to the fact that the image sources may be different. In addition, model inputs usually require fixed image sizes and formats. Therefore, we normalized each image. The first step was to resize the images, and all images were uniformly resized to a resolution of 224 × 224. This size is a common input requirement for deep learning models, which reduces the consumption of computational resources and retains sufficient feature information. Then the image format is converted and all images are unified to RGB format. Through these two operations, all the images in the dataset meet the requirements of the model in terms of size and format, avoiding problems due to inconsistent inputs during the training process. Finally, the normalization process, since the range of image pixel values is usually [0, 255], if directly input to the model, it may lead to problems such as unstable gradient or too small learning rate. Therefore, we normalize the pixel values of the image by scaling all the pixel values to the range [0, 1]. This operation is achieved by a simple mathematical transformation, i.e. dividing the pixel values by 255. Since we use autoencoder technology, there is no need to perform explicit feature selection. It indirectly achieves feature extraction and dimensionality reduction by automatically learning low-dimensional representations of normal data.

B. Model Development

We divide the model development into four steps, and the following are detailed explanations of the steps:

Step 1: Algorithm design

Existing methods have the following limitations: traditional statistical methods are difficult to process high-dimensional data, such as PCA and ARIMA, which rely on fixed data distribution and cannot adapt to dynamic environments; pure supervised learning methods rely on a large amount of labeled data, but abnormal data is often scarce in practical applications; using CAE alone may lead to a high false alarm rate, and slight changes in normal data may be mistakenly judged as abnormal based on reconstruction errors.

The current method (CAE-RF) is suitable for anomaly detection of autonomous driving camera sensors, mainly because: it can process high-dimensional, unstructured image data, CAE extracts deep features, retains spatial information, and is suitable for complex environments; it can detect known and unknown anomalies, and unsupervised CAE discovers unseen abnormal data through reconstruction errors, overcoming the dependence of traditional supervised learning on labeled data; it reduces the false alarm rate, and compared with methods that rely only on reconstruction errors, CAE-RF combines random forest classifiers to enhance the robustness of anomaly detection; it meets real-time requirements, and this method combines the feature extraction capabilities of deep learning with the efficient decision-making capabilities of random forests, which is suitable for the low latency requirements of autonomous driving systems.

Therefore, CAE-RF combines the generalization ability of unsupervised learning and the discrimination ability of supervised learning, overcoming the shortcomings of existing methods and becoming the best solution to the current problem.

Step 2: Autoencoder training

The training of the autoencoder is a key part of the development stage. In this stage, we use normal samples to train the autoencoder so that it can learn to reconstruct the distribution characteristics of normal samples. The structure of the autoencoder consists of an encoder and a decoder. The encoder compresses the high-dimensional image data into a low-dimensional potential feature space, while the decoder tries to reconstruct an image similar to the input data from the low-dimensional space. The model learns by minimizing errors between the input image and the reconstructed image through optimization of network parameters during training. Upon training, we can get the potential feature of normal data through the encoder, and calculate the reconstruction error of normal data through decoder.

Step 3: Feature extraction and calculation of reconstruction error

Feature extraction is an important step in model development. After the autoencoder is trained, the input data will generate potential features through the encoder part, and the decoder part will calculate the reconstruction error. The potential features are representative of the global properties of the input data and the reconstruction error is a measure of the extent to which the data deviates from the normal sample distribution. These two parts of the features are combined to form the final feature vector. Through this process, we convert the high-dimensional image data into a multi-dimensional feature space suitable for random forest training. This method not only effectively compresses the data, but also retains key abnormal information.

Step 4: Training of random forest model

The training of random forest model is the final link in the development stage. Based on the multi-dimensional features extracted by the autoencoder and the calculated reconstruction error, we use random forest to classify normal samples and abnormal samples. Random forest constructs multiple decision trees, and each tree independently learns the feature distribution of the data. During the training process, the model will continuously optimize the decision rules and ensure the stability and accuracy of the classification results through the majority voting mechanism.

Overall, the four links in the development stage are closely linked to form a complete system. The algorithm design provides a theoretical framework, the autoencoder training and feature extraction realize the acquisition of key features and the calculation of reconstruction errors, and the random forest model training transforms these features into efficient classification capabilities. This development process not only verifies the theoretical feasibility of the model, but also lays a solid technical foundation for subsequent system deployment.

C. Model Validation and Evaluation

Validation is essential to ensure that the model generalizes well to previously unseen data and performs reliably in a variety of scenarios. Use cross-validation to split the data set into training and testing subsets, allowing the model to be evaluated across multiple iterations. This approach mitigates over-fitting and ensures that the performance of the model does not depend on specific data partitions.

The evaluation phase uses a comprehensive set of metrics to measure the performance of the model. Accuracy assesses the proportion of correctly classified data points, while accuracy assesses the model's ability to avoid false positives. The recall rate determines how sensitive the model is to identifying real anomalies. The F1-score is a harmonic average of accuracy and recall, providing a balanced evaluation metric.

A comparative analysis is performed to compare the proposed framework with traditional methods such as statistical anomaly detection and time series analysis. These comparisons highlight the advantages of the hybrid approach, demonstrating greater accuracy, fewer false positives, and greater adaptability to complex scenarios.

The final stage involves systematically documenting the entire process to ensure that research methods, results and conclusions are clear, repeatable and available for future use. This stage integrates all aspects of the research into a coherent written record. It includes detailed descriptions of data preparation, model development and validation steps, ensuring transparency and enabling other researchers to replicate or build on them.

IV. Design and Experiments

A. *Autoencoder Model Design and Implementation*

In the task of anomaly data detection, we chose Convolutional Autoencoder (CAE) as the core tool for feature extraction. Compared with traditional deep autoencoders, convolutional autoencoders can process high-dimensional image data more efficiently, capture local features, and effectively preserve the spatial information of the input image. This section will describe the design and implementation process of the convolutional autoencoder in detail, including the model architecture, training methods, and practical applications in anomaly detection.

The structure of the convolutional autoencoder consists of two main parts: the encoder and the decoder, which are used for feature extraction and data reconstruction respectively.

The main task of the encoder is to compress the input image into a low-dimensional latent feature space while retaining the core information of the input data. Specifically, the encoder consists of a series of convolutional layers and pooling layers. These convolutional layers extract local features of the image, such as edges, textures, and shapes, through convolution kernels, while the pooling layers reduce the spatial resolution of the data by downsampling, thereby reducing computational complexity and avoiding overfitting. In this process, as the number of layers increases, the model gradually extracts higher-level abstract features and finally compresses the original image into a low-dimensional feature vector. The final output of the encoder is the representation of the input image in the latent space, which contains the core patterns and distribution of the input data.

The decoder is the symmetrical part of the encoder, and its task is to restore the low-dimensional latent feature vector to a reconstructed image with the same size as the input image. The decoder gradually increases the resolution of the feature map through deconvolution operations to restore the original spatial information. At the same time, the decoder also uses upsampling technology to enlarge the feature map through interpolation operations to approach the size and distribution of the original image. In the last layer of the decoder, by using the Sigmoid activation function, the model limits the pixel values of the reconstructed image to the range of [0, 1], which is consistent with the normalized input image. The design of the decoder complements the encoder. It forces the encoder to learn more representative latent features by minimizing the reconstruction error.

The loss function is the core of the convolutional autoencoder training process. Its role is to measure the difference between the input image and the reconstructed image and guide the parameter update of the model. We use the mean squared error (MSE) as the loss function, and its formula is as follows:

$$L = \frac{1}{N}\sum_{i=1}^{N}(x_i - \hat{x}_i)^2 \tag{1}$$

Where, x_i represents the pixel value of the original input image, $\hat{x}_i$ represents the pixel value of the reconstructed image, and N is the total number of pixels in the image. The mean squared error encourages the model to restore the original data as much as possible by quantifying the difference between the input image and the reconstructed image at the pixel level. By minimizing MSE, the encoder will learn the latent features that can efficiently represent the input image, and the decoder will optimize its restoration ability. In addition, another reason for choosing MSE as the loss function is its sensitivity to reconstruction error, which helps to distinguish normal samples from abnormal samples in anomaly detection tasks. Normal samples have low reconstruction errors because their distribution is fully learned by the model; however, abnormal samples have significantly higher reconstruction errors because they deviate from the normal distribution. This difference provides important clues for subsequent classifiers.

The following Fig. 3 shows the working structure of the autoencoder:

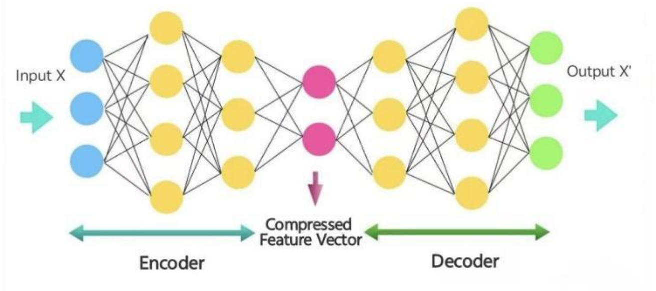

Fig. 3. Working structure of autoencoder.

B. *The Training Process of Convolutional Autoencoder*

In the training process of the convolutional autoencoder, we divide it into four main stages: data preparation, training configuration, model optimization, and model evaluation.

a) Data preparation: During autoencoder training, only normal samples are used to learn the distribution of normal data. Data preprocessing includes normalizing pixel values to [0, 1], resizing images to fit the model, and applying data augmentation techniques such as random rotation, noise addition, and brightness adjustment to enhance data diversity.

b) Training configuration: The input of the autoencoder is defined as (224, 224, 3), indicating that the processed image is an RGB image of 224*224 pixels. This input size is set by adjusting the input_shape parameter in the code. The encoder part uses two layers of convolutional layers and maximum pooling operations to gradually extract the core features of the image, and generates a 512-dimensional feature vector encoded through a fully connected layer. The decoder part restores the resolution and spatial structure of the image through dense connection layers, deconvolution and upsampling operations.

c) Model optimization: The Adam optimizer was selected when the model was compiled. The model can adaptively adjust the learning rate to accelerate the convergence process. The mean square error (MSE) was selected as the loss function to measure the pixel-level difference between the input image and the reconstructed image. The reason for choosing MSE is that it is very sensitive to reconstruction errors and can effectively capture the distribution deviation of abnormal data. During the training process, the input normal image is compressed into potential features by the encoder and then restored to the reconstructed image by the decoder. The model calculates the reconstruction error and continuously adjusts the parameters through back propagation to gradually reduce the reconstruction error. The number of training rounds is set to 20 (epochs=20) in the code, which is a reasonable value required for the model to converge.

d) Model evaluation: After training, we conducted a comprehensive evaluation of the model's performance, focusing on its performance on normal and abnormal data. First, we used normal samples and abnormal samples in the test set to calculate their reconstruction errors and analyze the difference in error distribution (Fig. 4) between the two types of data. The reconstruction error of normal samples is generally low, while the error of abnormal samples is significantly higher, which indicates that the model has successfully learned the distribution characteristics of normal data.

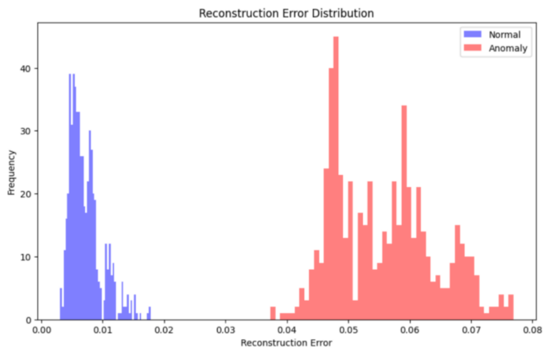

Fig. 4. Sample reconstruction error distribution.

C. Design and Implementation of Random Forest Model

RF is an ensemble learning algorithm for regression and classification [21-22]. During the training process, a large number of decision trees are constructed and the class of discrete tree output patterns is output [23]. It has strong robustness when dealing with high-dimensional features and multi-category classification tasks. Studies have shown that RF classifiers have superior performance compared to other methods such as neural network classifiers, bagging and boosting [24]. For classifying objects by image category, RF's performance is comparable to SVM, and RF has the advantage of being easier to train and test than SVM [25]. Taking the above factors into consideration, this paper designs and trains a random forest classifier by combining the reconstruction error calculated by the autoencoder and the hidden features extracted by the encoder to complete the binary classification task of normal and abnormal images.

The core idea of the random forest model is to form a strong classifier by constructing multiple weak classifiers (decision trees) and performing weighted voting on their prediction results, thereby improving the classification performance and generalization ability of the model. In this study, the input of the random forest classifier is the feature matrix of the image, and the features come from the reconstruction error calculated by the autoencoder and the hidden features extracted by the encoder. The input features consist of 512-dimensional hidden features and 1-dimensional reconstruction error, with a total feature dimension of 513. The hidden features capture the high-level semantic information of the image, while the reconstruction error reflects the degree of abnormality of the image. The number of decision trees (n_estimators), the maximum depth (max_depth), and the feature selection strategy (max_features) are important parameters of the random forest. In the experiment, these parameters are optimized by grid search to ensure that the model achieves a balance between classification performance and computational efficiency (Fig. 5).

The process for building an RF model with n decision trees can be summarized into three steps [26], as described below. Note that in the process, it is assumed that the data has k original features.

Step 1: Use the bagging method to generate n independent sample subsets from the initial data set.

Step 2: For each sample subset, build a classification or regression decision tree. When each node of the tree splits, randomly select k candidate features from all k features, and select the feature with the largest information gain as the split point. Finally, n decision trees will be generated.

Step 3: For classification tasks, integrate the prediction results of n trees by majority voting; for regression tasks, use the average value as the final output.

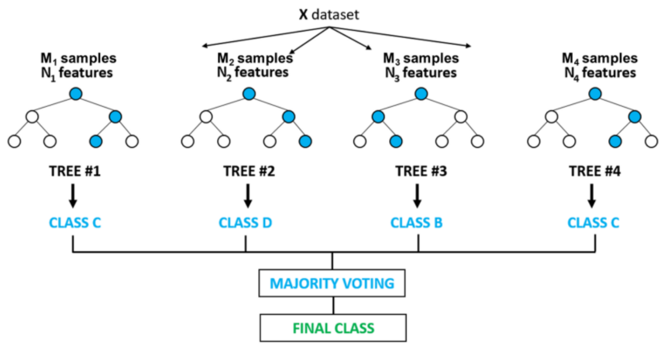

Fig. 5. Working structure of random forest.

D. Random Forest Training Process

a) Data preparation: In this study, the dataset consists of normal images and six types of abnormal images, including blur, noise, color shift, brightness, rotation, and occlusion. Each image is processed by the encoder to generate 512-dimensional hidden features, and the reconstruction error is calculated by the autoencoder. Finally, the dimension of the feature matrix is (number of samples, 513), and the corresponding label vector is (number of samples,). The ratio of normal images to abnormal images in the dataset is 1:1, and a total of 7024 images are included, with a balanced number of samples of normal images and each abnormal category. The dataset is divided into training set and validation set by stratified sampling, with the training set accounting for 80% and the validation set accounting for 20%. This partitioning method can ensure that the proportion of each category in the training set and validation set is consistent, thereby improving the generalization ability of the model.

b) Training configuration: The training parameters of the random forest classifier have an important impact on model performance and computational efficiency. This paper makes the following configurations based on input features and task requirements: First, the number of trees (n_estimators) is set to 200. Experiments show that increasing the number of trees can improve classification performance, but the computational time also increases. When the number of trees exceeds 200, the model performance tends to stabilize; second, the maximum depth (max_depth) is set to 20 to prevent a single decision tree from being too complex and causing overfitting, while controlling the training time; third, the Gini impurity (criterion) is used as the criterion for node splitting to ensure that each split can minimize the impurity of the category; finally, the feature selection strategy (max_features) is set to sqrt, that is, each time the split is performed, the square root of the features are randomly selected from all features for splitting to enhance the robustness of the model and reduce the training time.

c) Model optimization: The optimization of the random forest model is a systematic process that aims to improve the classification ability and generalization performance of the model by adjusting the model's hyperparameters and feature design. The optimization process mainly adjusts the four parameters in the previous section. For the number of decision trees, the range is set to [50, 100, 150, 200, 250, 300] through experiments. It is found that increasing the number of trees can improve the classification accuracy of the model, but the benefits decrease after exceeding a certain number. Finally, the number of trees is set to 200 to strike a balance between classification performance and training efficiency. In order to prevent overfitting caused by excessive growth of decision trees, the maximum depth is set to [10, 15, 20, 25] for testing. Experiments show that a depth of 20 can effectively control the complexity of the model while maintaining high classification performance. The classification criteria commonly used are Gini Impurity and entropy. After comparative testing, the results show that Gini Impurity has faster training speed and better classification performance in most categories. The feature selection strategy uses the default sqrt.

d) Model evaluation: The evaluation of the random forest model is mainly carried out by preliminary verification of the classification accuracy on the validation set to ensure that the model can effectively distinguish normal and abnormal images. The experimental results show that the classification accuracy rate reaches 92%. More specific classification results and analysis will be described in the next section.

V. Results and Analysis

A. Model Evaluation Methods

In this study, in order to verify the performance of our proposed model, we adopted common evaluation criteria [27-28], including classification accuracy, precision, recall, and F1 score.

The classification accuracy reflects the overall prediction accuracy of the model for all samples, and the calculation formula is:

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \tag{2}$$

Precision measures the proportion of samples predicted to be of a certain category that actually belong to that category. The calculation formula is:

$$Precision = \frac{TP}{TP+FP} \tag{3}$$

The recall rate measures the proportion of actual samples of a certain category that are correctly identified by the model. The calculation formula is:

$$Recall = \frac{TP}{TP+FN} \tag{4}$$

The F1 score is the harmonic average of precision and recall, and is used to comprehensively evaluate the classification performance of the model. In the case of unbalanced class samples, the F1 score can better reflect the actual classification effect of the model. The calculation formula is:

$$F1_{Score} = \frac{2*(Precision*Recall)}{Precision+Recall} \tag{5}$$

In the above formula, TP (true positive) represents the number of positive samples correctly classified, TN (tree negative) represents the number of negative samples correctly classified, FP (false positive) represents the number of positive samples incorrectly classified, and FN (false negative) represents the number of negative samples incorrectly classified.

B. Model Performance

In order to fully verify the performance of the model, we used a five-fold cross-validation method to conduct experiments. Specifically, we evenly divided the test dataset into five subsets. By rotating the test set, we ensured that each subset was used as a test set once and only once. Finally, we averaged the results of the five experiments to obtain a more robust performance evaluation.

TABLE I. RANDOM FOREST

Category	Test set	Accuracy	Precision	Recall	F1-score
Normal	1	0.90	0.88	0.92	0.900
	2	0.93	0.91	0.94	0.925
	3	0.89	0.87	0.91	0.890
	4	0.92	0.90	0.93	0.915
	5	0.91	0.89	0.92	0.905
	Ave	0.91	0.89	0.924	0.907
Abnormal	1	0.92	0.91	0.93	0.92
	2	0.94	0.93	0.95	0.94
	3	0.91	0.90	0.92	0.91
	4	0.95	0.94	0.96	0.95
	5	0.93	0.92	0.94	0.93
	Ave	0.93	0.92	0.94	0.93

As can be seen from Table Ⅰ, the five-fold cross-validation results of the CAE-RF model in the two major categories of "normal" and "abnormal". From the overall performance, the classification performance of the model in the "abnormal" category is better than that in the "normal" category, where the average precision and recall of the abnormal category reached 93% and 94% respectively, indicating that the model has high sensitivity and low missed detection rate when capturing abnormal samples. However, the precision of the "normal" category is slightly lower, only 89%, reflecting that the model occasionally misclassifies abnormal samples as normal samples. Overall, the model maintains good stability in classification performance, with an average F1 score of 90.7% (normal category) and 93% (abnormal category), providing reliable support for anomaly detection tasks in practical applications. In the future, the precision can be further improved by optimizing feature selection or adjusting hyperparameters.

C. Model Comparative Analysis

In order to verify the performance of the CAE-RF model, other classifiers (such as support vector machines, K nearest neighbors, deep neural networks, etc.) were introduced into the experiment for comparison, and all models were subjected to five-fold cross validation (Table II). The specific results are as follows:

TABLE II. MODEL COMPARISON

Model	Accuracy	F1-Score	Inference time/sample	Rank
KNN	82.63%	0.816	50ms	5
SVM	85.87%	0.868	30ms	4
Autoencoder	84.90%	0.852	2ms	3
RF	88.23%	0.879	10ms	2
CAE-RF	92.56%	0.933	5ms	1

2) K Nearest Neighbors (KNN): The Euclidean distance metric was used to select the optimal K value (K=5) through grid search and normalization was performed. The classification performance of KNN is limited by the distance measurement method under high-dimensional data, and the classification accuracy is low, with an average value of 82.63%. Since the inference stage needs to calculate the distance between each test sample and all training samples, the inference time is long (about 50 milliseconds/sample). KNN is suitable for small-scale data sets, but not suitable for real-time scenarios.

3) Support Vector Machine (SVM): The radial basis kernel function (RBF) was used with regularization parameter C=1.0 and kernel coefficient gamma=0.01, optimized by cross-validation. SVM outperforms KNN in classification performance, with an accuracy of 85.87%. However, SVM's inference time is too long at 30 milliseconds/sample, which limits its application in real-time tasks.

4) Autoencoder: The structure is the same as CAE (encoder 2 layers of convolution + pooling, decoder symmetric), training rounds 20, loss function MSE. The autoencoder performs anomaly detection by reconstructing the error, with a classification accuracy of 84.9%% and an average F1 score of 0.852. The inference speed is very fast, only 2 milliseconds/sample, which is suitable for unsupervised anomaly detection tasks, but the performance is limited when used alone.

5) Random Forest (RF): The number of decision trees is 200, the maximum depth is 20, and the feature selection strategy is square root (sqrt). Random Forest has achieved a good balance between classification performance and efficiency, with a classification accuracy of 88.23% and an average F1 score of 0.879. The inference time is only 10 milliseconds per sample, which is suitable for real-time classification tasks of high-dimensional feature data, and supports feature importance analysis, with a certain degree of interpretability.

6) CAE-RF (Convolutional Autoencoder + Random Forest): The CAE-RF model combines the feature extraction capability of the convolutional autoencoder with the robustness of the random forest, and performs best in classification performance, with an accuracy of 92.56% and an F1-score of 0.933 respectively. Its inference time is only 5ms, which is suitable for complex and real-time anomaly detection tasks.

VI. Discussion

This paper studies the application of a hybrid model based on autoencoder and random forest (CAE-RF) in image anomaly detection tasks. Traditional methods often have performance bottlenecks in high-dimensional data processing and classification tasks, and it is difficult to balance classification performance and real-time performance. To this end, this paper proposes an innovative feature fusion and classification framework, which extracts hidden features through autoencoders and combines them with reconstruction errors, and uses random forest classifiers to efficiently classify abnormal categories. Experimental results show that the CAE-RF model performs well in six categories of anomaly detection tasks and achieves a good balance between performance and efficiency.

The success of the CAE-RF model depends on the following key factors. First, the hidden feature extraction of the autoencoder significantly improves the expressiveness of the input features, while the reconstruction error further enhances the distinguishability of abnormal samples. This feature fusion method effectively makes up for the shortcomings of a single feature. Secondly, the robustness and interpretability of the random forest classifier provide the model with powerful classification capabilities, while the feature importance analysis also provides a transparent decision-making basis for the anomaly detection task. In addition, this paper verifies the significant performance advantages of the CAE-RF model through five-fold cross validation and multi-model comparison experiments, and proves its applicability in complex anomaly detection tasks.

However, despite its promising performance, the proposed method has certain limitations. The effectiveness of feature extraction relies on the autoencoder's reconstruction capability, which may be insufficient for detecting subtle or context-dependent anomalies, where abnormal features are not distinctly different from normal patterns. Furthermore, the limited size and diversity of the dataset could affect the model's generalization ability in real-world applications. Future work should explore larger and more diverse datasets, incorporate attention mechanisms or transformer-based architectures to enhance feature extraction, and investigate adaptive thresholding techniques to refine anomaly classification.

References

[1] Eskandarian, A., Wu, C., & Sun, C. (2019Research advances and challenges of autonomous and connected ground vehicles. IEEE Transactions on Intelligent Transportation Systems, 22(2), 683-711.

[2] Shladover, S. E. (2021). Opportunities and challenges in cooperative road vehicle automation. IEEE Open Journal of Intelligent Transportation Systems, 2, 216-224.

[3] Taiebat, M., Brown, A. L., Safford, H. R., Qu, S., & Xu, M. (2018). A review on energy, environmental, and sustainability implications of connected and automated vehicles. Environmental science & technology, 52(20), 11449-11465.

[4] Baccari, S., Hadded, M., Ghazzai, H., Touati, H., & Elhadef, M. (2024). Anomaly Detection in Connected and Autonomous Vehicles: A Survey, Analysis, and Research Challenges. IEEE Access.

[5] Ruff, L., Kauffmann, J. R., Vandermeulen, R. A., Montavon, G., Samek, W., Kloft, M., ... & Müller, K. R. (2021). A unifying review of deep and shallow anomaly detection. Proceedings of the IEEE, 109(5), 756-795.

[6] Erhan, L., Ndubuaku, M., Di Mauro, M., Song, W., Chen, M., Fortino, G., ... & Liotta, A. (2021). Smart anomaly detection in sensor systems: A multi-perspective review. Information Fusion, 67, 64-79.

[7] Chandola, V., Banerjee, A., & Kumar, V. (2009). Anomaly detection: A survey. ACM computing surveys (CSUR), 41(3), 1-58.

[8] Hodge, V., & Austin, J. (2004). A survey of outlier detection methodologies. Artificial intelligence review, 22, 85-126.

[9] Markou, M., & Singh, S. (2003). Novelty detection: a review—part 1: statistical approaches. Signal processing, 83(12), 2481-2497.

[10] Mohamudally, N., & Peermamode-Mohaboob, M. (2018). Building an anomaly detection engine (ADE) for Iot smart applications. Procedia computer science, 134, 10-17.Chandola, V., Banerjee, A., & Kumar, V. (2009). Anomaly detection: A survey. ACM computing surveys (CSUR), 41(3), 1-58.

[11] Mani, I., & Zhang, I. (2003, August). kNN approach to unbalanced data distributions: a case study involving information extraction. In Proceedings of workshop on learning from imbalanced datasets (Vol. 126, No. 1, pp. 1-7). ICML..

[12] Aggarwal, C.C., & Zhai, C. (2012). Mining Text Data. Springer US.

[13] Ting, K. M. (2002). An instance-weighting method to induce cost-sensitive trees. IEEE Transactions on Knowledge and Data Engineering, 14(3), 659-665.

[14] Weiss, G. M., & Provost, F. (2003). Learning when training data are costly: The effect of class distribution on tree induction. Journal of artificial intelligence research, 19, 315-354.

[15] Han, Y., Xu, M., & Guan, L. (2024, April). Conformalized semi-supervised random forest for classification and anomalyity detection. In International Conference on Artificial Intelligence and Statistics (pp. 2881-2889). PMLR.

[16] Joshi, M. V., Agarwal, R. C., & Kumar, V. (2001, May). Mining needle in a haystack: classifying rare classes via two-phase rule induction. In Proceedings of the 2001 ACM SIGMOD international conference on Management of data (pp. 91-102).

[17] Lee, W., Stolfo, S. J., Chan, P. K., Eskin, E., Fan, W., Miller, M., ... & Zhang, J. (2001, June). Real time data mining-based intrusion detection. In Proceedings DARPA Information Survivability Conference and Exposition II. DISCEX'01 (Vol. 1, pp. 89-100). IEEE.

[18] Rajasegarar, S., Leckie, C., Palaniswami, M., & Bezdek, J. C. (2006, October). Distributed anomaly detection in wireless sensor networks. In 2006 10th IEEE Singapore international conference on communication systems (pp. 1-5). IEEE.

[19] Fiore, U., Palmieri, F., Castiglione, A., & De Santis, A. (2013). Network anomaly detection with the restricted Boltzmann machine. Neurocomputing, 122, 13-23.

[20] Garcia-Font, V., Garrigues, C., & Rifà-Pous, H. (2016). A comparative study of anomaly detection techniques for smart city wireless sensor networks, 16(6), 868.

[21] Breiman, L. (2001). Random forests Mach Learn 45 (1): 5–32.

[22] Ho, T. K. (1995, August). Random decision forests. In Proceedings of 3rd international conference on document analysis and recognition (Vol. 1, pp. 278-282). IEEE.

[23] Biau, G., Devroye, L., & Lugosi, G. (2008). Consistency of random forests and other averaging classifiers. Journal of Machine Learning Research, 9(9).

[24] Ham, J., Chen, Y., Crawford, M. M., & Ghosh, J. (2005). Investigation of the random forest framework for classification of hyperspectral data. IEEE Transactions on Geoscience and Remote Sensing, 43(3), 492-501.

[25] Bosch, A., Zisserman, A., & Munoz, X. (2007, October). Image classification using random forests and ferns. In 2007 IEEE 11th international conference on computer vision (pp. 1-8). IEEE.

[26] Niaf, E., Rouvière, O., Mège-Lechevallier, F., Bratan, F., & Lartizien, C. (2012). Computer-aided diagnosis of prostate cancer in the peripheral zone using multiparametric MRI. Physics in Medicine & Biology, 57(12), 3833.

[27] Wang, L., You, Z. H., Xia, S. X., Liu, F., Chen, X., Yan, X., & Zhou, Y. (2017). Advancing the prediction accuracy of protein-protein interactions by utilizing evolutionary information from position-specific scoring matrix and ensemble classifier. Journal of Theoretical Biology, 418, 105-110.

[28] Gao, Z. G., Wang, L., Xia, S. X., You, Z. H., Yan, X., & Zhou, Y. (2016). Ens-PPI: A Novel Ensemble Classifier for Predicting the Interactions of Proteins Using Autocovariance Transformation from PSSM. BioMed research international, 2016(1), 4563524.