Intelligent Automation & Soft Computing

Tech Science Press

DOI: 10.32604/iasc.2023.040310
Article

Abnormal Behavior Detection Using Deep-Learning-Based Video Data Structuring

Min-Jeong Kim[1], Byeong-Uk Jeon[1], Hyun Yoo[2] and Kyungyong Chung[3,*]

[1]Department of Computer Science, Kyonggi University, Suwon-si, 16227, Gyeonggi-do, Korea
[2]Contents Convergence Software Research Institute, Kyonggi University, Suwon-si, 16227, Gyeonggi-do, Korea
[3]Division of AI Computer Science and Engineering, Kyonggi University, Suwon-si, 16227, Gyeonggi-do, Korea

*Corresponding Author: Kyungyong Chung. Email: dragonhci@gmail.com

Received: 13 March 2023; Accepted: 10 May 2023; Published: 23 June 2023

Abstract: With the increasing number of digital devices generating a vast amount of video data, the recognition of abnormal image patterns has become more important. Accordingly, it is necessary to develop a method that achieves this task using object and behavior information within video data. Existing methods for detecting abnormal behaviors only focus on simple motions, therefore they cannot determine the overall behavior occurring throughout a video. In this study, an abnormal behavior detection method that uses deep learning (DL)-based video-data structuring is proposed. Objects and motions are first extracted from continuous images by combining existing DL-based image analysis models. The weight of the continuous data pattern is then analyzed through data structuring to classify the overall video. The performance of the proposed method was evaluated using varying parameter settings, such as the size of the action clip and interval between action clips. The model achieved an accuracy of 0.9817, indicating excellent performance. Therefore, we conclude that the proposed data structuring method is useful in detecting and classifying abnormal behaviors.

Keywords: Deep learning; object detection; abnormal behavior recognition; classification; data structuring

1 Introduction

Internet-of-Things-based devices, black boxes, and closed-circuit televisions (CCTVs) generate vast amounts of video data. Currently, approximately 1.3 million CCTVs are installed in domestic public institutions, and the number is increasing every year [1]. Although it cannot be accurately measured, the number of CCTVs used in private properties is estimated to be much higher than that in public institutions. Owing to the large amounts of data collected by CCTVs, it is infeasible for supervisors to check every video [2]. Although supervisors must check these videos for public safety, there are concerns regarding privacy violations in the process. Because abnormal behaviors comprise a small fraction of a footage, identifying these behaviors manually is highly cumbersome. CCTVs are typically used to detect crimes in real time; however, repeated monitoring is required for confirmation after a crime has occurred, as the abnormal behavior in question must not only be detected but also classified. Deep-learning (DL)-based research is being actively conducted to detect objects and recognize actions from video data [3], with algorithms such as you look only once (YOLO) [4] and faster region-based convolutional neural network (R-CNN) [5] being used for object extraction. Inferring the relationship between objects by detecting them makes it possible to determine the characteristics of a video. However, it is difficult to discriminate between images using only object information. For example, if objects such as elephants, giraffes, and zebras are extracted from an image, it can be inferred that the image category is "animals". However, if the extracted information consists entirely of people, it is difficult to categorize the image because of the level of ambiguity between objects. To address this, motion analysis methods, such as SlowFast networks, or motion extraction algorithms based on the human skeleton can be used to classify human motion [6]. However, because these techniques extract only simple motions based on short video clips, it is difficult to use them to analyze abnormal behaviors that occur on a scale of tens of seconds.

This study proposes an abnormal behavior detection method using DL-based video data structuring. The proposed method extends the common DL-based feature extraction method to combine object and behavior information in the form of a transaction. Objects and behaviors were extracted using YOLO and SlowFast networks, and the extracted data were structured using the bow method. Because the data structure consisted of numerous object and behavior columns, blank columns were deleted during preprocessing to prevent the emergence of a sparse matrix. Finally, each instance of abnormal behavior was classified using an automated machine learning (autoML) model suitable for structured data. Thus, it is possible to detect abnormal behaviors by analyzing object/behavior patterns from image data. Furthermore, accurate and fast prediction becomes possible by structuring the data using a machine learning model that is lighter than DL, which enables the accurate detection of abnormal situations and quickly provides relevant information. Our main contributions are as follows:

- Video is converted into structured data through the proposed data structuralize, and based on this, it is possible to detect abnormal behavior situations according to object and behavioral information.
- It is possible to use a pre-trained action recognition model and has high versatility as it does not require additional learning according to data.
- It is possible to effectively analyze a large amount of video data. After data structurization, the data does not require much computing power for subsequent processing. This enables effective analysis of large amounts of video data.
- Consequently, Using the data structuring proposed in this study, higher video behavior classification accuracy is shown.
- In addition, it is possible to classify abnormal behaviors of various types.

2 Related Work

2.1 Deep-Learning-Based Video Analysis

Existing object detection algorithms largely utilize the R-CNN series [7], which has a very slow detection speed-approximately 5 s per frame. Even Faster R-CNN [8], the fastest among R-CNN algorithms, has a maximum detection speed of seven frames per second, making it inefficient for image processing. In contrast, YOLO [9] features an object detection speed of 45 fps. In the case of R-CNN, the region of interest in which the object is expected to exist is first extracted through the region proposal network (RPN). Thereafter, classification is performed using a classifier for each region of interest, and a bounding box is determined. In comparison to R-CNN series models composed of RPNs and class classifiers, YOLO maintains high accuracy in object detection and quickly improves the speed by simplifying the network structure. However, it has the limitation of being unable to detect multiple small objects gathered in a single grid.

In motion detection, the use of temporal information is crucial. It is possible to store and use past information using an algorithm based on recurrent neural networks [10]. However, these algorithms are limited by memory decay. To resolve this, Long Short Term Memory (LSTM) [11] was designed to use only short-term memory for learning, and a convolutional LSTM [12] model that implements a CNN to reflect spatial features in images was additionally proposed. This method accounts for both temporal and spatial features, whereas the original LSTM model cannot reflect spatial features. Zhu et al. proposed a skeleton-based motion recognition method that uses a bidirectional LSTM-CNN model [13]. This method extracts the relative position and velocity information through the skeleton sequence, and subsequently enters a Bi-LSTM model. The output of the model is then input into the CNN classifier to classify the motion. However, there is a limitation-the images taken from three different angles are required when constructing the dataset. Feichtenhofer et al. proposed SlowFast networks [14], which exhibit excellent performance in motion recognition classification by mimicking the human ocular system. Under this model, slow motion has a different contribution level than fast motion during video recognition. Unlike its identity, the motion of objects experiences rapid changes. Accordingly, the spatial structures and temporal events of each object should be considered separately. The SlowFast networks comprises two pathways that extract frames at different rates. Semantic information is captured through a slow pathway that requires fewer frames per second, whereas motion information is captured through a fast pathway that requires a larger number of frames per second. Finally, features derived from each pathway are combined and classified. Although each pathway was configured based on the same CNN network, the number of channels in each kernel was set differently. With fewer channels, spatial modeling performance decreases, whereas temporal modeling ability increases. The amount of computation is reduced, making the model lightweight. Accordingly, the fast pathway was configured with fewer channels than the slow pathway. The calculation ratio between the slow and fast pathways was set to 8:2, equivalent to the ratio of M-cells to P-cells in the human eye. Like their corresponding pathways, these cells respond to temporal and spatial information, respectively. Thus, the SlowFast networks functions as a two-stream net that mimics the human visual perception system. Although the model performed excellently in the action classification of the Kinetics-400 dataset, this dataset consists of images in which human figures appear relatively large. In contrast, most abnormal behaviors recorded on CCTV videos show people in small sizes, which limits the applicability of the SlowFast networks to CCTV footage. Sun et al. proposed multistream SlowFast-graph convolutional networks (SF-GCN) for skeleton-based action recognition [15]. The model consists of six SF-GCN models: three models that consider the joints in the skeleton sequence and three models that consider the edges. Each of these models receives sequences of spatial and temporal differences as input. Final predictions are made by combining the features derived from each SF-GCN model.

Xiong et al. designed a transferable two-stream convolutional network for human action recognition [16]. This network optimizes weights after the pretraining phase, which uses large-scale open-source data. Subsequently, the layer transfer is performed such that the pretrained model can recognize behavior in the target domain, where the learning data are limited. This process enables the use of temporal information while mitigating data restrictions. Chen et al. proposed a heterogeneous two-stream network [17], where temporal and spatial information is extracted via ResNet and batch normalization inception, respectively. To obtain long-range temporal information, the video is segmented, trained, and averaged to derive the behavior class score. Likewise, in conventional methods, action recognition tasks are generally performed by receiving video data as input. Fig. 1 shows the architecture of the action recognition task used in prior studies. However, background information is not considered because the action recognition is performed by extracting only the human skeleton, and the objects in the vicinity are not considered [18].

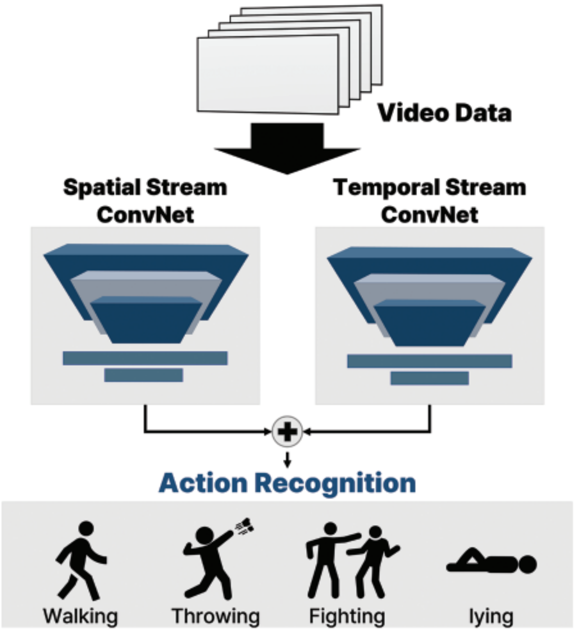


Figure 1: Architecture of the action recognition task used in prior studies

2.2 Semantic Extraction and Extension Using Text Mining

Because most data in the current age is unstructured, the analysis of unstructured data has become increasingly valuable [19]. Text mining is an analytical process that derives meaningful information for various purposes according to statistics or linguistics [20]. This approach is often used for tasks related to the classification or clustering of texts by identifying the word or topic characteristics inherent to a document. Text mining is also used to extract specific entities, or keywords, from within a document, making the method applicable to tasks involving unstructured text. Text mining is performed in the following order: First, unstructured data are collected and a vocabulary is extracted through processes such as morphological analysis. Necessary information is extracted from the vocabulary before analysis. To analyze the text, it is first converted into vector form, where each word is transformed into either local or distributed representation [21]. Local representation maps words to specific values, such as one-hot vectors, N-grams, or bag of words (BoW) [22], whereas distributed representation expresses continuous values by considering surrounding words, as in latent semantic analysis and Glove [23].

In this study, meaningful patterns were extracted through text mining using DL-based image analysis. In most action-recognition tasks, object actions are classified based on temporal and spatial information of the video data. However, few studies classified actions after structuring of unstructured data. In this study, DL-based image-analysis models were employed to extract objects as nouns and actions as verbs, thus generating a data structure. In the preprocessing stage, extraneous objects and behavioral variables were removed using the word representation method. The preprocessed data were learned using autoML, and an ensemble model was used to classify abnormal behaviors.

3 Abnormal Behavior Detection Using Deep-Learning-Based Video Data Structuring

As the amount of video data generated by CCTVs increases, the need to collect and analyze images also increases. Image analysis research is generally conducted by learning object motion using a CNN-based DL model and determining abnormal behaviors through a process equivalent to binary classification. Rather than simply identifying abnormal behaviors in a binary fashion, it is necessary to classify these behaviors spatially as well as temporally. Therefore, we propose an abnormal behavior detection method using DL-based video data structuring, where image patterns are identified according to the passage of time.

Fig. 2 shows the proposed classification method that consists of three stages. In the object detection stage, an object is detected and tracked using the YOLO model. In the action classification stage, the action is recognized using the tracked person object as a SlowFast networks input. Finally, the abnormal behavior classification stage employs the autoML model to classify abnormal behaviors from structured data.

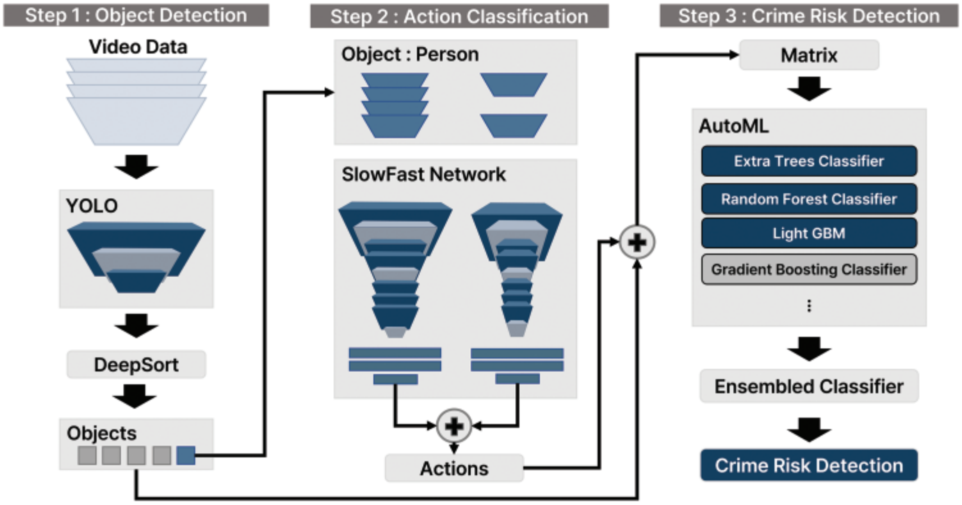


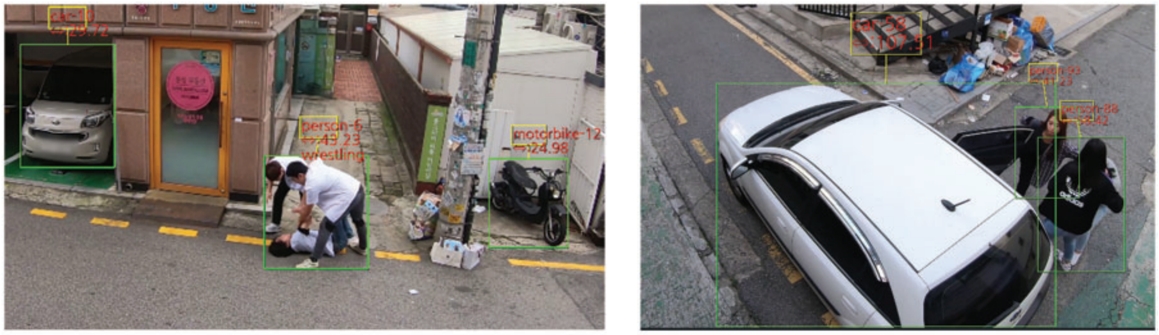
Figure 2: Process of abnormal behavior detection using deep-learning-based video data structuring

3.1 Collection and Preprocessing of Time Series Data

The YOLO system and DeepSORT are used for object detection and object tracking, respectively. DeepSORT is an object tracking framework that complements simple online and real-time tracking (SORT) [24]. First, the location of an object is predicted and updated in the next frame using an object that appeared previously through Kalman filtering and by measuring the intersection over union (IOU). In the next step, IOU similarity is measured through IOU match, and objects are classified. Objects to be classified are previously tracked objects and untracked objects that disappear or newly appear in the next frame. Objects in progress are updated with measurement values through the Kalman filter in the same way as in the first step, and objects that disappear from the frame are deleted after some time. In addition, a new track is created and added in the case of a newly appearing object. Next, a human motion recognition algorithm is used for action recognition. This study employed

SlowFast networks, a DL recognition model based on the structure of ocular cells. Compared to the conventional structure in which the image and optical flow are entered into 3D-ConvNet in parallel for learning and then summed, the proposed model simultaneously takes two images as input. Using the YOLO and SlowFast networks, an object is detected through the above process, and the behavior of the object is recognized.

Fig. 3a shows the result according to the fight class data, and Fig. 3b shows the result according to the kidnap class data. For each object detected in the Fig. 3, its type, relative distance to the camera, and behavior are displayed in a yellow box, along with a corresponding bounding box outlined in green. Information regarding objects is recorded in the log for every frame, and all objects and actions in the video are extracted and stored in text form composed of nouns and verbs.



(a) Fight Class Data

(b) Kidnap Class Data

Figure 3: Object detection and action recognition results using YOLO and SlowFast networks

Table 1 shows object matrices corresponding to the first stored video frame. Each matrix comprises the object name, unique identification number, Cartesian coordinates, dimensions, relative distance to camera, and behavior. When multiple objects appear in a single frame, they are allocated into separate rows. When the same object is added multiple times, a unique ID is assigned to each object to distinguish it. Because the number of objects appearing in each frame varies, the length of the matrix differs for each frame. Subsequently, an object-action frequency matrix for each frame is constructed to extract object-behavioral patterns according to time. In the original CSV file, only the object and behavior variables are used, excluding all other variables. These data are then preprocessed and inputted into the autoML model for learning.

Table 1: First matrix according to video frame

Frame	Obj.	Obj. ID	X	Y	Width	Height	Dist.	Motion	...
6750	Person	520	2421	845	280	376	36.42		...
6751	Person	519	2584	947	206	444	33.94	Parkour	...
6752	Person	519	2571	946	211	452	34.66	Parkour	...
6753	Person	520	2431	866	273	380	36.15		...
6754	Person	520	2426	901	274	442	37.06		...
6755	Person	520	2426	907	269	441	38.66		...
⋮									

3.2 Data Structuring

To enable data structuring, features must be extracted based on the frequency of each variable in the initial matrix, according to the video frame in Table 1. To extract features, a BoW structure [22] was used. The BoW method is frequently used in natural language processing to express words based on frequency, regardless of order. Fig. 4 shows the data structuring process under the BoW method.

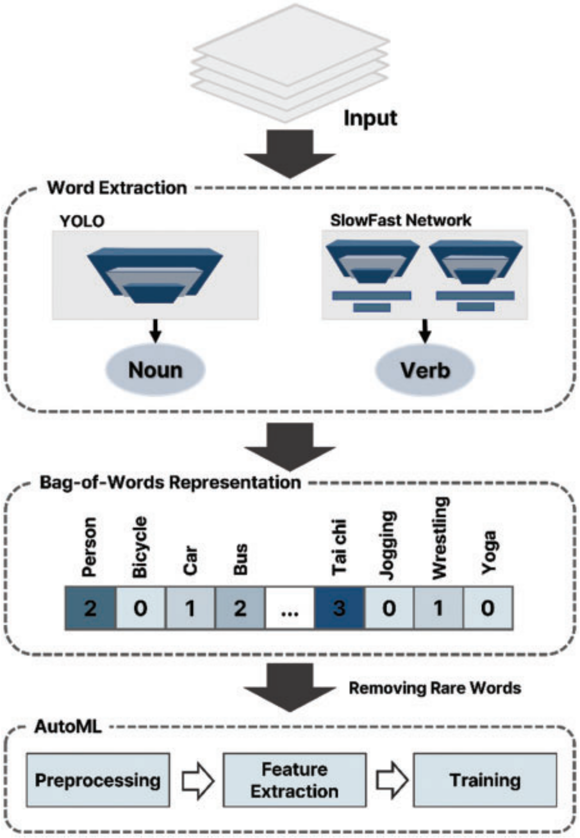


Figure 4: Object detection and action recognition results using YOLO and SlowFast networks

In Fig. 4, the image data is input into the YOLO and SlowFast networks to extract object nouns and action verbs, respectively. Each extracted word becomes a column, and the BoW structure expresses the frequency of all words appearing in each frame. For each extracted word, preprocessing is performed by removing words that appear less frequently in the video. The preprocessed data are then input into autoML for learning. In the BoW technique, each video can be considered a document, each frame a sentence. Accordingly, a frequency matrix of objects and behavior variables is created for each frame. The frequency matrix consists of 480 columns, including 80 objects of coco data and 400 actions of kinetic_400 data.

Algorithm 1 shows pseudocode for matrix generation. When features are vectorized through BoW, sparse matrix data are generated when missing objects or behaviors are filled with 0. However, a preprocessing stage is required because sparse data generally degrades the model performance [25]. Moreover, CCTV footage always includes objects and behaviors that have little correlation with abnormal behavior; therefore, any variables with a frequency of zero are removed to avoid the sparse matrix problem.

Algorithm 1: Pseudocode for matrix generation

```
FUNCTION matrix_generator (index)
    csv_file = []
    OPEN file specified by index
    READ contents into csv_file
    count_matrix = []
    FOR EACH row in csv_file DO
        line = []
        FOR EACH item in the row DO
            IF item is a string containing only alphabetical characters THEN
                APPEND item to line
            END IF
            JOIN items in line with a space character and assign to lines
        END FOR
        APPEND lines to count_matrix
    END FOR
    result = []
    FOR EACH row in count_matrix DO
        APPEND an empty list to result
        d = row as string
        FOR EACH item in obj_act_list DO
            t = item as string
            APPEND the count of t in d to the last list in result
        END FOR
    END FOR
    final_matrix = CREATE pandas DataFrame using result as data and obj_act_list as column names
    SAVE final_matrix as CSV file named matrix_path + 'matirx_' + file_list_csv [index]
END FUNCTION
```

Table 2 shows a frequency matrix consisting of 60 objects and behavioral variables, excluding variables with little correlation. Through the behavior frequency matrix, it is possible to determine a video pattern according to the passage of time. One such behavior frequency matrix was generated for each video.

Table 2: Behavior frequency matrix of video data

Frame	Person	Bicycle	Car	...	Tai chi	Yoga	...
4105	3	0	0	...	1	0	...
4106	3	0	0	...	1	0	...
4107	3	0	0	...	1	0	...
4108	3	0	0	...	0	1	...
			⋮				

In Table 2, a person is shown as an object in Frame 4105, and tai chi behaviors were detected between human objects. This suggests that the behavior labeled tai chi occurred through the collective interaction between three people. In addition, because the matrix comprises a series of frames, it is possible to compare objects and behavior patterns before and after an abnormal behavior occurs. Because not all frames from the video data are related to abnormal behavior, it is necessary to extract specific frames in which abnormal behavior appears using the XML file of the video. To extract the name, start time, and duration of an abnormal event from the XML file, all event names and action tags are parsed. Subsequently, any frames exhibiting continuous abnormal behavior are extracted.

For general CCTV analysis, the Abnormal behavior CCTV video dataset was provided by the AI Hub [26], an open data site for AI training, jointly run by the Republic of Korea Ministry of Science and Information and Communications Technology (ICT), and the Korea Intelligence Information Society Promotion Agency. This website provides AI training data owned by domestic and foreign institutions and companies for small- and medium-sized venture companies, research institutes, and individuals that generally have difficulty securing high-quality, large-capacity training data. The dataset in question was constructed using the Data Construction Project for Artificial Intelligence Learning of the Ministry of Science and ICT of the Republic of Korea and the Korea Intelligence Information Society Promotion Agency. It is comprised of CCTV videos exhibiting 12 types of abnormal behaviors, such as assault, fight, and burglary, with a total of 8,436 MP4 files. Each video file is associated with an XML context definition file that functions as a label for information about the video, including resolution and channel information. The header area contains information regarding the length of the video, frames per second, total number of frames, indoor/outdoor classification, weather, and number of appearing objects. It also includes the classification and duration of the event shown in the video. The position of each object is represented by x and y coordinates, whereas the duration of each event is measured in frames.

A dictionary is employed to label the 12 types of abnormal behaviors, with the behavior name as the key and an integer value from 1 to 12 as the corresponding value. This value is inserted into the label of the start and end frames corresponding to the event. To adjust the 32-frame delay in the result of the SlowFast networks, 32 frames are removed from the first frame in which the abnormal behavior is labeled before preprocessing. The accuracy of the model can be verified by comparing the degree of matching between the prediction results of the autoML model and the context definition XML data.

3.3 Abnormal Action Classification Using AutoML

AutoML is a process that enables the automated development of machine learning models [27]. Previously, to implement a model through machine learning, a developer would have to select an appropriate algorithm and set the parameters to optimize the model. Because training multiple machine learning models incurs high resource and time costs, automated learning is crucial. Consequently, autoML is applied not only in regression and classification, but also in computer vision and natural language processing. All necessary parameters are set before learning, and preprocessing is performed according to the nature of the training data. In the learning process, autoML selects the appropriate algorithm and learns the optimal hyperparameters. Subsequently, models with excellent performance in various metrics are aligned. Fig. 5 shows the overall autoML process for the abnormal action classification.

The Pycaret python library was used to construct autoML for anomaly classification [28]. We use 10 classes: fight, burglary, vandalism, swoon, wander, trespass, dump, robbery, kidnap, drunken. With the addition of a normal behavior classification, 11 classes were considered prior to the classification. Before the training phase, appropriate parameters were set for autoML. First, the variable fix_imbalance method for the data setup was set to random under sampler.

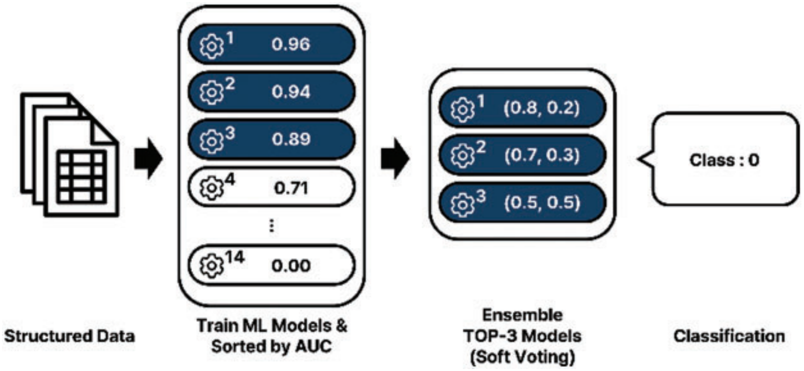
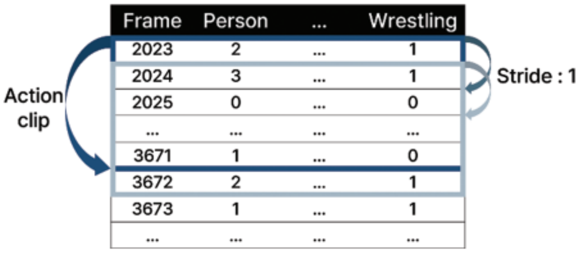


Figure 5: Process of autoML for abnormal action classification

Random under sampling is a method of randomly sampling data belonging to each class [29], to ensure variable performance. When sorting the models, the performance index of each model was designated as the area under the ROC curve (AUC) [30], and the final model was derived by an ensemble of the top three models. The verification technique for the ensemble uses k-fold [31], which is a cross-validation method that divides the data in the fold into k subsets, allocates k−1 subsets as training data, and uses the remaining subset as validation data. When the value of k increases, the time required for training also increases; therefore, it is necessary to set an appropriate value. In this experiment, the value of k is set to 5. Soft voting was used as the voting method for the ensemble models. This method determines the class by adding the class determination probabilities of each model and averaging them [32]. Because abnormal behavior occurs over multiple frames, continuous frames corresponding to an abnormal event are bundled into one action clip for training purposes. Each action clip consists of 2n frames like 128, 256, and 512. Some action clips are derived by generating an action clip for each frame at certain intervals in a single video. Stride, which is a hyperparameter that determines how many intervals to configure for the action clips, is set to multiples of 5, that is, 1, 5, 10, and 20. Fig. 6 shows the concepts of action clips and strides. In Fig. 6, the action clip consists of 128 frames, and the stride is 1 because there is a single-frame interval.



Frame	Person	...	Wrestling
2023	2	...	1
2024	3	...	1
2025	0	...	0
...	...	...	...
3671	1	...	0
3672	2	...	1
3673	1	...	1
...	...	...	...

Figure 6: Overview of action clips and strides

After training, results for 14 classification models, including logistic regression, K-neighbors classifier, and decision tree classifier, are obtained. Among these models, the extra tree classifier is a tree model that randomly subdivides each feature, whereas the random forest classifier is an ensemble model that trains and synthesizes several decision tree models [33]. The gradient boosting classifier employs boosting and carries out learning using the residual fitting method [34]. The light gradient boosting machine is a transformation model of gradient boosting that minimizes loss by creating an unbalanced tree through a leaf-wise structure. The AdaBoost classifier is an algorithm that improves the performance of the final classifier by increasing the weight of improperly classified data [35]. Logistic regression is an algorithm that classifies data by predicting the probability that it belongs to a specific category, and the K-neighbors classifier is an algorithm that classifies data using the properties of the adjacent data based on the distance between data [36]. The decision tree classifier is a tree-structured supervised-learning-based model that classifies data features [37]. Naïve Bayes is a statistical classification model that employs the Bayes theorem [38]. The quadratic and linear discriminant analysis algorithms are probabilistic generative models that find the probability distribution of y with respect to x through Bayes theorem. The dummy classifier is a classifier model that makes predictions without trying to find patterns in data. The base model identifies the most frequent labels in the training dataset and makes predictions based on those labels. If the model performance is similar to that of the dummy classifier, it is considered that model learning has not occurred. If the model performance is greater than that of the dummy classifier, it is considered that the reliability of the model has been secured. In this way, the baseline is specified by comparing the performance of classification models.

SVM–linear kernel is a model that defines an optimal decision boundary for data classification [39]. The ridge classifier is a model that performs the classification task through ridge regression after transforming data into values between -1 and 1.

4 Experiments

A pre-trained YOLOv5 model was used for object detection, and a pre-trained SlowFast networks with kinetic-400 data was used for action recognition. AutoML was used to classify abnormal behaviors. To evaluate the accuracy of the classification model, input data were constructed according to the size of the action clip and stride. Of the total data, 15.4 h of videos for each class were used, with 70% of the data being allocated to training and verification, and the remaining 30% allocated to testing. The hardware used to implement the model consisted of an Intel® Xeon® Silver 4210R, 128 GB, and NVIDIA GeForce RTX 3090. The model was built using Python 3.8.8, PyTorch 1.7.1, and CUDA 11.0.

The following evaluation metrics were used:

(1) Accuracy: The percentage of correct answers for the entire evaluation results based on the confusion matrix, which compares predicted values with actual values. Due to its intuitiveness, accuracy is the most common performance metric. Eq. (1) is the formula for calculating accuracy using the confusion matrix.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{1}$$

Here, true positive (TP) indicates a match between the actual and predicted values, and true negative (TN) means that an incorrect answer has been predicted as an incorrect answer. A false positive (FP) is a case where a false value was predicted as true, whereas a false negative (FN) indicates that a true result was predicted as false.

(2) AUC: AUC refers to the area below the ROC curve. It has a value between 0 and 1; with higher values indicating better performance.

(3) Recall: This represents the percentage of the correct values that have been predicted to be true by the classification model.
(4) Precision: Precision represents the ratio of values that are correct to those predicted as true by the classification model.
(5) F1-score: As the harmonic average of precision and recall, the F1-score can accurately measure the performance of a model when the data class has an unbalanced structure.
(6) Kappa: The probability that the interpretations between different observers coincide, expressed as a value between 0 and 1.
(7) Matthews correlation coefficient (MCC): The quality of a classification model, measured using a confusion matrix. It is expressed as a value between −1 and +1, where higher values are associated with correct predictions.

4.1 Evaluation of Abnormal Behavior Classification

The first evaluation assessed the performance of abnormal behavior classification by combining various models through the autoML process. For a total of 14 classifiers, three evaluation metrics—accuracy, AUC, and precision—were measured and compared. Table 3 shows the performance results for various classifiers for action clips with a stride of 1.

Table 3: Performance comparison result (The action clip is 256 and stride is 1)

Model	Acc.	AUC	Recall	Prec.	F1	Kappa	MCC
Extra trees classifier	0.9794	0.9993	0.9139	0.9795	0.9785	0.9749	0.9750
Random forest classifier	0.9744	0.9992	0.8954	0.9748	0.9731	0.9689	0.9690
Light gradient boosting machine	0.9685	0.9990	0.8657	0.9683	0.9669	0.9617	0.9618
Gradient boosting classifier	0.9031	0.9893	0.7520	0.9026	0.8975	0.8812	0.8823
Logistic regression	0.8923	0.9864	0.7362	0.8893	0.8887	0.8689	0.8692
K neighbors classifier	0.8932	0.9821	0.7977	0.9032	0.8948	0.8712	0.8719
Decision tree classifier	0.9324	0.9607	0.8249	0.9317	0.9313	0.9189	0.9180
Quadratic discriminant analysis	0.6354	0.8169	0.4849	0.9848	0.7453	0.5789	0.6294
Naïve Bayes	0.6588	0.8154	0.5574	0.7344	0.6877	0.5985	0.6019
Linear discriminant analysis	0.6971	0.8078	0.4988	0.7481	0.7148	0.6372	0.6393
Ada boost classifier	0.4401	0.7322	0.2833	0.4197	0.4042	0.3099	0.3170
Dummy classifier	0.3108	0.5000	0.0909	0.0966	0.1474	0.0000	0.0000
SVM—Linear kernel	0.8483	0.0000	0.6746	0.8536	0.8459	0.8160	0.8169
Ridge classifier	0.7605	0.0000	0.5431	0.7641	0.7575	0.7092	0.7100

Under the proposed method, the extra tree classifier produced the best performance in all evaluation metrics, followed by the random forest classifier. Although these classifiers are similar, they feature differences in data sampling and feature selection methods: extra trees, unlike random forests, do not use bootstrap sampling, instead, they use sampling without replacement; thus, they use the entire data. The random forest method calculates and compares the information gain for all variables in feature selection to find the optimal feature value. However, extra trees split nodes by randomly extracting features from all variables. Because the features are randomly partitioned, extra trees quickly appear. Random forest achieved high performance for the validation set because many trees were used in the ensemble, thus reducing bias and variance.

4.2 Comparison of Accuracy According to Classification Parameters

The second evaluation compared accuracy according to the action clip and stride value. We use an ensemble of the three models with the highest AUC derived from autoML for classification. Fig. 7 shows the results of an accuracy comparison according to the action clip and stride.

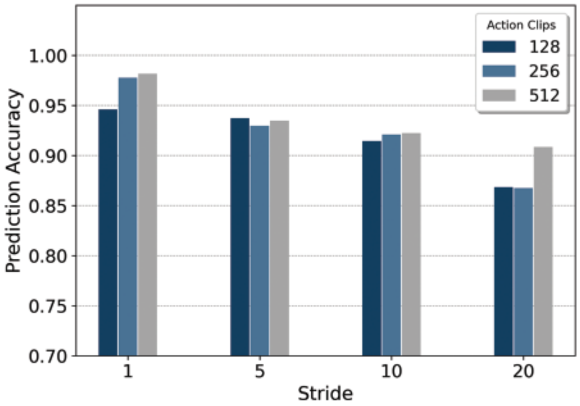


Figure 7: Accuracy comparison results according to frame and stride

Here, the highest accuracy was obtained for the 512-frame clips with a stride length of 1. Furthermore, the 512-frame clips were associated with the highest accuracy measures regardless of stride length, with the difference in accuracy ranging from 0.007 to 0.04. This is explained by the fact that larger clips contain more extensive object and action information and encompass a longer time.

Likewise, a stride of 1 maximized the number of frames shown in each video, thus ensuring the highest accuracy. The difference in accuracy with respect to stride length ranged from 0.073–0.083. We note that the difference in accuracy due to stride length was greater than that with respect to action clip size. Therefore, we can conclude that while both metrics are significant, the stride length has a greater impact on performance. For comparison experiments, video classification is performed using a CNN-LSTM-based model. ResNet-based CNN and VGG-based CNN models are used for comparative experiments. Due to memory limitation, the input videos are resized to 160 × 160 and set action clips to 40. The batch size is set to 3, and epochs are continued until the validation loss does not decrease. Table 4 shows the performance comparison between the CNN-LSTM series model [40] and our model proposed in this paper.

Compare the accuracy, AUC, and F1 score of each model. The performance of CNN (ResNet50)-LSTM model and CNN (VGG19)-LSTM model is lower than our model. Our model performed better than other models, recording an accuracy of 0.9814, an AUC of 0.9998, and an F1 score of 0.9793. In the case of CNN-LSTM, the video itself is input. Accordingly, it is impossible to consider high-resolution and long frames at once due to memory limitations. Therefore, compared to the model proposed in this paper, fewer action clips and resolution are used, resulting in reduced accuracy.

Table 4: Performance comparison with CNN-LSTM model

Method	Accuracy	AUC	F1
CNN (ResNet50)-LSTM [40]	0.3708	0.6599	0.3148
CNN (VGG19)-LSTM [40]	0.7615	0.8698	0.7020
Proposed	**0.9814**	**0.9998**	**0.9793**

5 Conclusions

In many cases that require the identification of abnormal behavior from CCTV footage, the footage in question is not appropriately used. Therefore, there is an increasing need to recognize objects and actions portrayed in these videos. In this study, we proposed an abnormal behavior detection method using DL-based video data structuring to analyze the patterns of objects and behaviors in video data. In this method, data related to objects and motion are extracted using a DL-based image analysis algorithm for data structuring. The objects and events that appear in each frame are analyzed using the BoW technique, and preprocessing is performed to eliminate sparse data from the matrix. Each temporal frame is parsed for abnormal behavior in the continuous video data. A follow-up processing method using autoML classifies a total of 10 abnormal and normal behavior types. The performance of the proposed method was evaluated in two ways. The first experiment evaluated the validation results of the model based on 14 classifiers. In the second test, the accuracy of the model was measured with respect to the action clip size and stride length. The results of the performance evaluation produced a high accuracy measure of 0.9817. These results indicate that the proposed method successfully classifies abnormal behavior by identifying patterns in object and behavior data, enabling the recognition and dissemination of information regarding dangerous behavior. Thus, a large amount of CCTV video data can be analyzed effectively using this method. Our study derived high accuracy after data structuring. However, there is a limitation in that a lot of computing resources are consumed in structuring the video. Therefore, we plan to conduct research on efficient data structuring as a future study.

Funding Statement: This research was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education (2020R1A6A1A 03040583).

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

[1] Statistics Korea, 2022. [Online]. Available: http://kostat.go.kr/

[2] V. Singh, S. Singh and P. Gupta, "Real-time anomaly recognition through CCTV using neural networks," *Procedia Computer Science*, vol. 173, pp. 254–263, 2020.

[3] H. B. Zhang, Y. X. Zhang, B. Zhong, Q. Lei, L. Yang *et al.,* "A comprehensive survey of vision-based human action recognition methods," *Sensors*, vol. 19, no. 5, pp. 1005, 2019.

[4] J. Redmon, S. Divvala, R. Girshick and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, Silver Spring, MD, USA, pp. 779–788, 2016.

[5] S. Ren, K. He, R. Girshick and J. Sun, “Faster r-cnn: Towards real-time object detection with region proposal networks,” in *Advances in Neural Information Processing Systems*, Montreal, Canada, vol. 28, 2015.

[6] J. Carreira and A. Zisserman, “Quo vadis, action recognition a new model and the kinetics dataset,” in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, Honolulu, HI, USA, pp. 6299–6308, 2017.

[7] S. S. Park, J. W. Baek, S. M. Jo and K. Chung, “Motion monitoring using mask R-CNN for articulation disease management,” *Journal of the Korea Convergence Society*, vol. 10, no. 3, pp. 1–6, 2019.

[8] Z. He and L. Zhang, “Multi-adversarial faster-rcnn for unrestricted object detection,” in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, Seoul, Korea, pp. 6668–6677, 2019.

[9] J. Redmon and A. Farhadi, “Yolov3: An incremental improvement,” arXiv preprint arXiv: 1804.02767, 2018.

[10] W. Mao, M. Wang, W. Sun, L. Qiu, S. Pradhan *et al.,* “Rnn-based room scale hand motion tracking,” in *Proc. of the 25th Annual Int. Conf. on Mobile Computing and Networking*, Los Cabos, Mexico, pp. 1–16, 2019.

[11] Y. Yu, X. Si, C. Hu and J. Zhang, “A review of recurrent neural networks: LSTM cells and network architectures,” *Neural Computation*, vol. 31, no. 7, pp. 1235–1270, 2019.

[12] X. Shi, Z. Chen, H. Wang, D. Y. Yeung, W. K. Wong *et al.,* “Convolutional LSTM network: A machine learning approach for precipitation nowcasting,” in *Advances in Neural Information Processing Systems*, Montreal, Canada, 2015.

[13] A. Zhu, Q. Wu, R. Cui, T. Wang, W. Hang *et al.,* “Exploring a rich spatial-temporal dependent relational model for skeleton-based action recognition by bidirectional LSTM-CNN,” *Neurocomputing*, vol. 414, pp. 90–100, 2020.

[14] C. Feichtenhofer, H. Fan, J. Malik and K. He, “Slowfast networks for video recognition,” in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, Seoul, Korea, pp. 6202–6211, 2019.

[15] N. Sun, L. Leng, J. Liu and G. Han, “Multi-stream SlowFast graph convolutional networks for skeleton-based action recognition,” *Image and Vision Computing*, vol. 109, pp. 104141, 2021.

[16] Q. Xiong, J. Zhang, P. Wang, D. Liu and R. X. Gao, “Transferable two-stream convolutional neural network for human action recognition,” *Journal of Manufacturing Systems*, vol. 56, pp. 605–614, 2020.

[17] E. Chen, X. Bai, L. Gao, H. C. Tinega and Y. Ding, “A spatiotemporal heterogeneous two-stream network for action recognition,” *IEEE Access*, vol. 7, pp. 57267–57275, 2019.

[18] H. Yoo, R. C. Park and K. Chung, “IoT-based health big-data process technologies: A survey,” *KSII Transactions on Internet and Information Systems (TIIS)*, vol. 15, no. 3, pp. 974–992, 2021.

[19] K. Adnan and R. Akbar, “An analytical study of information extraction from unstructured and multidimensional big data,” *Journal of Big Data*, vol. 6, no. 1, pp. 1–38, 2019.

[20] S. Qaiser and R. Ali, “Text mining: Use of TF-IDF to examine the relevance of words to documents,” *International Journal of Computer Applications*, vol. 181, no. 1, pp. 25–29, 2018.

[21] Z. Liu, Y. Lin and M. Sun, “Representation learning and NLP,” in *Representation Learning for Natural Language Processing*. Singapore: Springer, pp. 1–11, 2020.

[22] H. D. Abubakar, M. Umar and M. A. Bakale, “Sentiment classification: Review of text vectorization methods: Bag of words, Tf-Idf, Word2vec and Doc2vec,” *SLU Journal of Science and Technology*, vol. 4, no. 1&2, pp. 27–33, 2022.

[23] J. Du, P. Jia, Y. Dai, C. Tao, Z. Zhao *et al.,* “Gene2vec: Distributed representation of genes based on co-expression,” *BMC Genomics*, vol. 20, no. 1, pp. 7–15, 2019.

[24] N. Wojke, A. Bewly and D. Paulus, “Simple online and realtime tracking with a deep association metric,” in *Proc. 2017 IEEE Int. Conf. on Image Processing (ICIP)*, Beijing, China, pp. 3645–3649, 2017.

[25] M. J. Kim, “T5-based video captioning for abnormal behavior detection,” M.S. Thesis, Department of Computer Science, Kyonggi University, Suwon-Si, South Korea, 2022.

[26] AI Hub, 2019. [Online]. Available: https://aihub.or.kr/

[27] X. He, K. Zhao and X. Chu, "AutoML: A survey of the state-of-the-art," *Knowledge-Based Systems*, vol. 212, pp. 106622, 2021.

[28] U. Gain and V. Hotti, "Low-code AutoML-augmented data pipeline–a review and experiments," *Journal of Physics: Conference Series*, vol. 1828, no. 1, pp. 012015, 2021.

[29] N. Habib, M. Hasan, M. Reza and M. M. Rahman, "Ensemble of CheXNet and VGG-19 feature extractor with random forest classifier for pediatric pneumonia detection," *SN Computer Science*, vol. 1, no. 6, pp. 1–9, 2020.

[30] H. Jung and K. Chung, "Social mining based clustering process for big-data integration," *Journal of Ambient Intelligence and Humanized Computing*, vol. 12, no. 1, pp. 589–600, 2021.

[31] H. L. Vu, K. T. W. Ng, A. Richter and C. An, "Analysis of input set characteristics and variances on k-fold cross validation for a recurrent neural network model on waste disposal rate estimation," *Journal of Environmental Management*, vol. 311, no. 114869, pp. 1–10, 2022.

[32] S. Kumari, D. Kumar and M. Mittal, "An ensemble approach for classification and prediction of diabetes mellitus using soft voting classifier," *International Journal of Cognitive Computing in Engineering*, vol. 2, pp. 40–46, 2021.

[33] N. Dogru and A. Subasi, "Traffic accident detection using random forest classifier," in *Proc. 2018 15th Learning and Technology Conf.*, Jeddah, Saudi Arabia, pp. 40–45, 2018.

[34] R. Punmiya and S. Choe, "Energy theft detection using gradient boosting theft detector with feature engineering-based preprocessing," *IEEE Transactions on Smart Grid*, vol. 10, no. 2, pp. 2326–2329, 2019.

[35] P. Bahad and P. Saxena, "Study of adaboost and gradient boosting algorithms for predictive analytics," in *Proc. Int. Conf. on Intelligent Computing and Smart Communication 2019: Proc. of ICSC 2019*, Singapore, pp. 235–244, 2020.

[36] K. Shah, H. Patel, D. Sanghvi and M. Shah, "A comparative analysis of logistic regression, random forest and KNN models for the text classification," *Augmented Human Research*, vol. 5, no. 1, pp. 1–16, 2020.

[37] Priyanka and D. Kumar, "Decision tree classifier: A detailed survey," *International Journal of Information and Decision Sciences*, vol. 12, no. 3, pp. 246–269, 2020.

[38] C. K. Aridas, S. Karlos, V. G. Kanas, N. Fazakis and S. B. Kotsiantis, "Uncertainty based under-sampling for learning naive Bayes classifiers under imbalanced data sets," *IEEE Access*, vol. 8, pp. 2122–2133, 2019.

[39] J. Cervantes, F. Garcia-Lamont, L. Rodríguez-Mazahua and A. Lopez, "A comprehensive survey on support vector machine classification: Applications, challenges and trends," *Neurocomputing*, vol. 408, pp. 189–215, 2020.

[40] A. M. R. Abdali and R. F. Al-Tuma, "Robust real-time violence detection in video using cnn and lstm," in *Proc. 2019 2nd Scientific Conf. of Computer Sciences (SCCS)*, Baghdad, Iraq, pp. 104–108, 2019.