Exploring the potential of skeleton and machine learning in classroom cheating detection

Nha Tran[1], Minh Nguyen[1], Tri Le[1], Tuong Huynh[1], Tinh Nguyen[1], Tuan Nguyen[2]
[1]Faculty of Information Technology, University of Education, Ho Chi Minh City, Vietnam
[2]Department of Information Technology, University of Education, Ho Chi Minh City, Vietnam

Article Info

Article history:

Received Jun 10, 2023
Revised Aug 8, 2023
Accepted Sep 3, 2023

Keywords:

Cheating detection
Long short-term memory
Machine learning
Skeleton
YOLO-Pose

ABSTRACT

In recent years, there have been certain drawbacks in the behavior of students in school cultural activities, especially the frequent occurrence of cheating in exams. Although many manual cheating detection methods have been applied, cheating behaviors have become increasingly sophisticated and difficult to detect. This makes manual cheating detection methods unable to completely prevent cheating in exams. Therefore, applying advanced technologies to automatically detect cheating cases becomes necessary to help supervisors better control the exam process. In this study, we propose a model architecture to monitor students' cheating activities during exams. Firstly, we apply YOLO-Pose to detect skeleton keypoints, which are then passed through machine learning models such as support vector machine (SVM), decision tree (DT) classifier, random forest (RF), extreme gradient boosting (XGBoost), and propose the Ac Long short-term memory (LSTM) to detect cheating behavior. The experimental results show the feasibility of the model on various metrics. The model is tested on a dataset of cheating behaviors in the classroom, which is designed to simulate paper-based exams. Moreover, the results showed that the proposed method can detect cheating behavior in a short amount of time and can be deployed for real-world applications.

Corresponding Author:

Nha Tran
Faculty of Information Technology, University of Education
280 An Duong Vuong St., Ward 4, District 5, Ho Chi Minh City, Vietnam
Email: nhatt@hcmue.edu.vn

1. INTRODUCTION

Cheating in examinations is an act of dishonesty during the process of conducting a test, which includes the use of unfair ways to achieve a favorable outcome. A number of employees at a private institution in the Philippines participated in a survey and admitted to having cheated at least once during their high school education, despite knowing that cheating is inappropriate [1]. According to a research conducted by King *et al.* [2] about the attitudes and behaviors of economics students towards cheating in an online exam, 73.6% of respondents believed that online cheating is an easy task without the fear of being detected. The study named "Contract cheating and the market in essays" [3] asked the students allowed to decide whether to cheat or not in the low detection condition. It showed that 50% of those students agreed to engage in cheating behavior. This evidence indicates that cheating detection methods must be highly accurate and effective in order to make a fair exam. Besides that, this research also pointed out the limited human resources of manual monitoring methods in large-scale contests. Examinations occur in an environment that is not conducive to detecting every gesture and action of individuals in the examination room. These reasons lead to a reduction in fairness preservation. Therefore, advanced cheating detection methods with high accuracy and the ability to handle large scales are required to effectively prevent cheating behavior.

Action recognition in videos has been a popular and effective topic in computer vision. However, this field also poses several challenges due to factors such as frame brightness, occluded objects, and camera motion during video recording. These challenges affect the ability to recognize actions [4]. Action recognition based on the 3D skeleton is a widely explored topic and has been extensively used in various fields such as human-computer interaction, autonomous driving, virtual reality, and more. The skeleton provides efficient information about the motion patterns of objects and has the advantage of requiring less computational resources compared to RGB and optical flow data. Additionally, it is less influenced by changes in lighting conditions and background noise [5].

Several studies have proposed different approaches to address the issue of cheating detection, aiming to achieve better results than manual methods. Nishchal *et al.* [6] proposed to use OpenPose to detect subject postures, then the Alexnet model to identify cheating actions, facial recognition, and emotion analysis. This method has a weakness in terms of data, as the recorded actions are captured from only fixed-angle cameras. Xing and Zhu [5] created a database from recordings during live exams. The model's performance was evaluated by detecting cheating actions in each frame, resulting in high accuracy. However, it is challenging to implement the system in practical applications due to its complexity in processing.

In recent years, numerous researchers have focused on describing and constructing cheating detection systems for exams, with the aim of applying them in real-world scenarios. Soman *et al.* [7] proposed a method based on K-nearest neighbors (KNN) and histogram of oriented gradients (HOG) to extract image features, achieving an accuracy of 88.2% for normal, turning around, passing paper, peeping, and signaling behaviors. As the study used the histogram of oriented gradients method to count the occurrence of gradient orientations in the video frames to detect object actions, there are some obstacles if multiple test takers are sitting close to each other and obstructing the view, as taking the parameters of a single frame would make the subsequent classification process more complex.

Sharma *et al.* [8] designed a model using convolutional neural network (CNN) to detect cheating behavior by analyzing the facial expressions of candidates during online exams. The accuracy of the false accept rate (FAR) and false reject rate (FRR) calculations have been tested and proven to yield excellent results in various lighting and facial expression conditions. This method effectively captures the facial expressions of candidates through a front-facing camera positioned at their seating location. However, this approach can be costly as each camera may require significant investment, especially for in-person exams. For online exams, candidates would need to have a camera to participate. Arinaldi and Fanany [9] utilized a 3DCNN model, XGBoost, and long short-term memory (LSTM) to process action sequences and textually describe videos on a dataset of 71 cheating videos, including actions such as document exchange, looking at nearby individuals, opening documents, and talking. The gesture recognition model achieved an accuracy score of 81.11%, while the language model utilizing LSTM achieved a score of 95.3%. The system ran at a speed of 32.54 FPS on a mid-range laptop. One limitation of the experiment was that the model could only detect up to 2 students in the frame.

The study by Li *et al.* [10] employed the RAE algorithm along with the expectation-maximization (EM) algorithm to assess students' cognitive levels, and coupled it with LSTM to determine their level of knowledge mastery based on their previous problem-solving history in exams. They then incorporated information on students' cognition levels, seating arrangements in the exam room, answer-guessing habits, exam question similarities into the model to evaluate whether a student was cheating or not. Experimental results showed that this method achieved significantly higher accuracy and recall rates than some commonly used methods. However, relying solely on capacity diagnosis may result in false positives for students who have the potential to improve their skills. Bancud [11] has developed a system to support the monitoring of live exams by determining the posture of students that indicates cheating. With an accuracy of 90%, F1-score of 89.65%, and AUROC of 90.32%, the monitoring system uses the 25 keypoint model of OpenPose and detects cheating using the gradient boosting technique with the XGBoost library for real-time performance and maximum accuracy with less than 10 FPS.

In addition to cheating in online exams, there are also many research works that identify direct anomalous behaviors in the classroom. Malhotra and Chhabra [12] suggested an automatic system for recognizing anomalous behaviors. They used YOLOv3 [13] combined with ShuffleNet [14] and achieving an average precision (AP) measure of 88.03% for the detection of cheating in the classroom environment. This model only identifies the initial part of the object to detect cheating, which may miss many body movements of the object. Sangalli *et al.* [15] suggested a cheating detection method that consists of two stages, reviewing course materials and solving exercises by students. K-means clustering is used to differentiate students who use fake accounts to collect correct answers for exercises. A generalization accuracy of over 95% to classify types of cheating is achieved by using support vector machine (SVM).

In this research, we propose a model architecture to detect cheating activities of students during exams. Firstly, we apply YOLO-Pose to detect skeletal keypoints, which are then passed through machine learning models such as SVM, DT classifier, RF, XGBoost, and propose the Ac-LSTM model to detect cheating behavior. The model is experimented on a dataset of cheating behaviors in a classroom, which is designed to simulate paper-based exams. By employing skeleton recognition as a foundational component for classification models, we aim to streamline the complexity of the system and enhance its practicality compared to previously published research on cheating detection in exams.

2. DATA PREPARATION AND ACQUISITION

2.1. Data preparation process

Due to the sensitivity and privacy concerns, most datasets related to cheating in paper-based exams are not publicly available. Therefore, we have designed and prepared a simulated dataset for paper-based exams. A paper-based exam is a centrally organized test where students come to the test room under the supervision of invigilators. This dataset includes various behaviors that students can take to cheat during the exam. We hope that this dataset will become a valuable resource for research and the development of methods to prevent cheating in paper-based exams. The data preparation process is described in Figure 1.

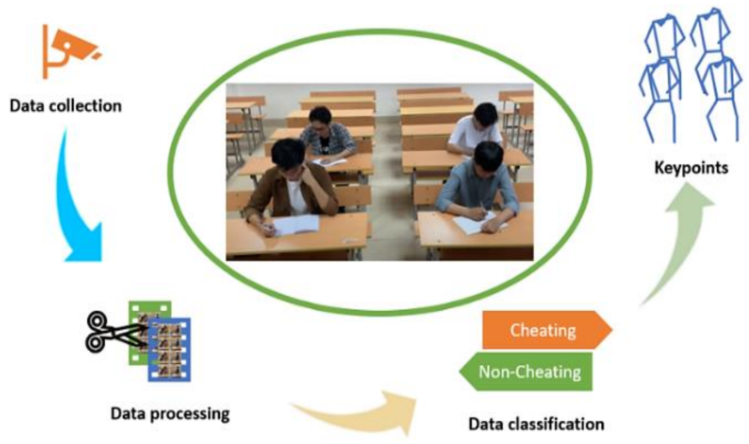


Figure 1. Data preparation process

2.2. Data acquisition

The number of participants involved is 12 individuals. The participants are provided with a consent form upon arrival at the data collection site and are required to provide consent for the use of their personal images in the data collection process. Prior to data acquisition, they are thoroughly explained about the data collection procedure and the purpose of data gathering. The detailed explanation of the data collection procedure and its purpose ensures transparency and the participants' consent regarding the use of their personal images.

The scene takes place in a classroom with three main angles: left side, right side, and frontal view. Additionally, the distance between the camera angles and the subject is 3 meters to ensure that the examinee's behaviors can be observed clearly and accurately. This is done to replicate the positioning of cameras arranged in a classroom setting. These camera angles provide comprehensive coverage of the entire classroom, ensuring the objectivity of the data collection process. The camera angles are illustrated in Figure 2.

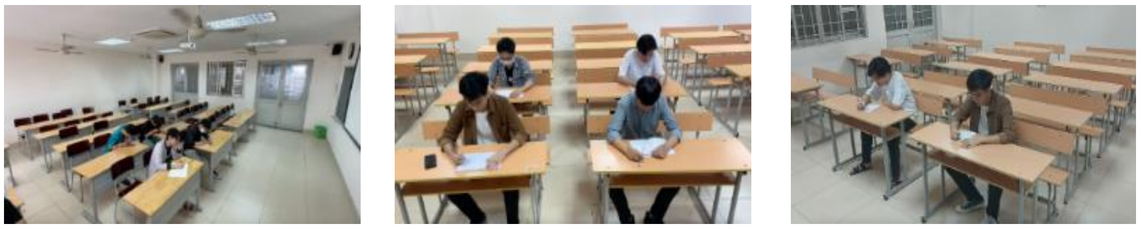


Figure 2. The camera angles in the exam room

2.3. Data processing

The dataset consists of 6 videos collected in the initial stage, with a total file size of 3.6 gigabytes. The videos were captured from different angles: 2 videos from the left angle, 2 videos from the right angle, and 2 videos from the frontal view. Each video has an average duration of about 15 minutes, comprising two main segments: the first segment captures the examinee's behavior without any signs of cheating, and the second segment records the examinee's behavior after they have started cheating in Figure 3.

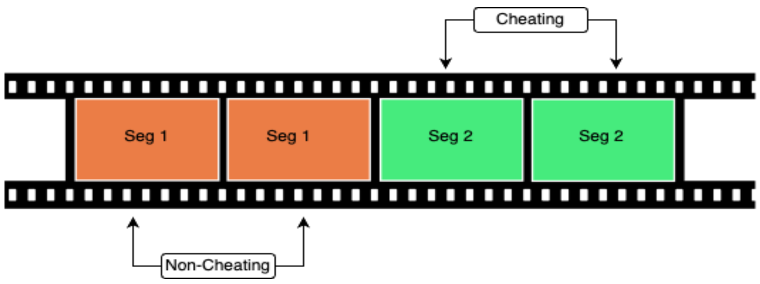
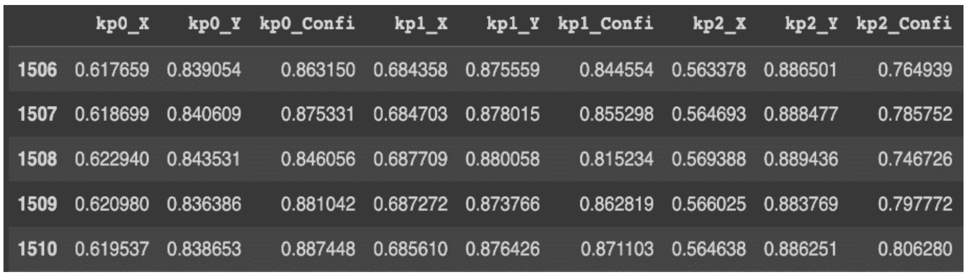


Figure 3. Segments in the video

The dataset needs to be processed to prepare it for the classfication stage before being fed into the training model. Firstly, all videos were synchronized to ensure consistency and resolution quality, with a resolution of 1280×720 for each video. Then, the audio was removed to reduce the video file size and focus solely on the examinee's actions, as the model only utilizes video frames. This data processing step is crucial to ensure the quality and accuracy of the model. To continue the data clasification process, we proceeded to segment the videos based on the examinee's behavior. The total number of collected videos was 710, with each video having a duration of approximately 4-6 seconds and containing a varying number of frames ranging from 100 to 130 frames. These videos were then classified into two main categories: "non-cheating" and "cheating." Specifically, there were 310 videos associated with "cheating" behaviors and 400 videos representing "non-cheating" behaviors. Within the "cheating" category, there were specific behaviors such as using reference materials, exchanging exam papers, copying answers, looking at other examinees' papers, and so on.

2.4. Data labeling

After completing the behavior classification, we proceed to collect the parameters of the skeleton using YOLO-Pose [16] YOLO-Pose: "enhancing YOLO for multi person pose estimation using object keypoint similarity loss," YOLO-Pose: "enhancing YOLO for multi person pose estimation using object keypoint similarity loss,". The skeletal parameters include 18 key points as shown in Figure 4. These skeletal parameters are then stored in a file in comma-separated values (CSV) format. Storing the data in CSV format allows researchers to easily share the data with other collaborators or access the data in different machine learning models.



	kp0_X	kp0_Y	kp0_Confi	kp1_X	kp1_Y	kp1_Confi	kp2_X	kp2_Y	kp2_Confi
1506	0.617659	0.839054	0.863150	0.684358	0.875559	0.844554	0.563378	0.886501	0.764939
1507	0.618699	0.840609	0.875331	0.684703	0.878015	0.855298	0.564693	0.888477	0.785752
1508	0.622940	0.843531	0.846056	0.687709	0.880058	0.815234	0.569388	0.889436	0.746726
1509	0.620980	0.836386	0.881042	0.687272	0.873766	0.862819	0.566025	0.883769	0.797772
1510	0.619537	0.838653	0.887448	0.685610	0.876426	0.871103	0.564638	0.886251	0.806280

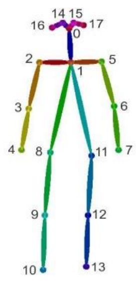


Figure 4. The CSV format structure file

In the CSV file, each row of the table represents a skeleton frame, and each keypoint contains the corresponding parameters (x, y, conf). Here, (x, y) represents the coordinates of the keypoints on the frame, and conf denotes the confidence score of the model in recognizing the keypoint. Additionally, an extra column labeled "label" is included to determine the posture of each action as either "cheating" or "non-cheating".

The keypoints are numbered and stored in the order corresponding to the columns of the table in the CSV file. To gain a better understanding of the cheating actions described in the skeletal frames, Figure 5 provides a detailed description of the keypoints and postures associated with "cheating" behaviors. The keypoints are used to determine the curvature of the body, the tilting of the head, and other body parts, thereby aiding in the accurate and effective detection of cheating behaviors.

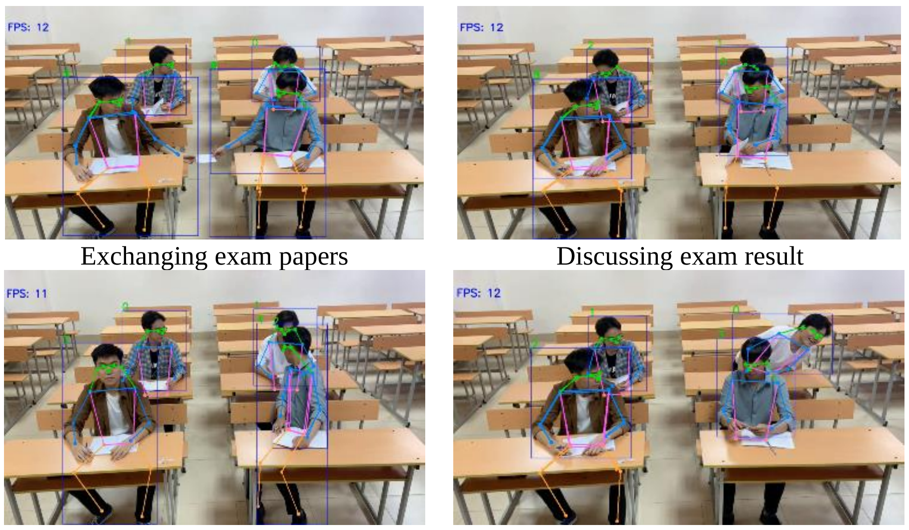


Figure 5. The cheating behaviors are described using skeletal frames

3. PROPOSED METHOD

In this research, we propose to build a real-time automatic detection model for cheating behavior by applying the YOLO-Pose model [16] to recognize skeletal frames of multiple objects, combined with machine learning models to classify cheating behaviors, as illustrated in Figure 6. First, we apply the YOLO-Pose model to the videos to estimate the poses for the person in each frame. However, for object detection models, there is an issue with the bounding boxes of each object not being ordered in the frame sequence. Deal with the problem, we use the intersection over union (IoU) algorithm to compare the bounding boxes of objects across consecutive frames. The next step is to take the keypoint parameters of the action sequence in the frames and input them into commonly used machine learning models such as SVM [17], DT [18], RF [19], and XGBoost [20]. Additionally, we also build a deep learning model using LSTM network to recognize the actions of the students across action sequences.

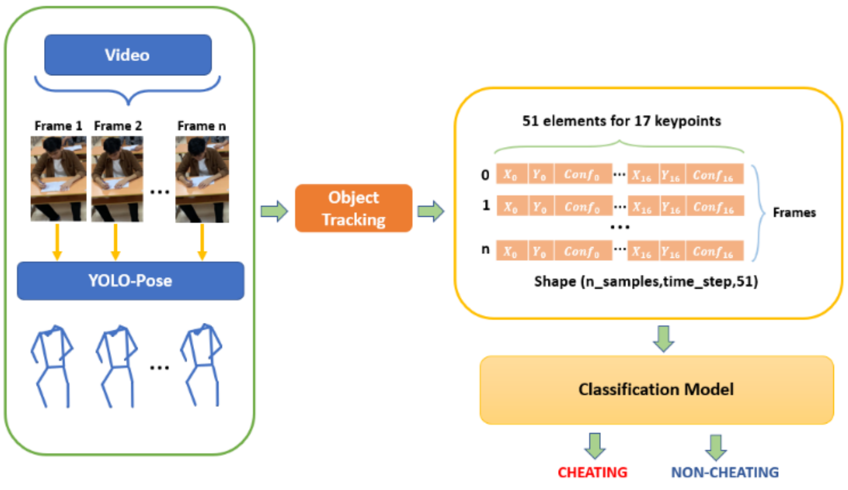

Video
Frame 1
Frame 2
Frame n
YOLO-Pose
Object Tracking
51 elements for 17 keypoints
0
1
n
X_0 Y_0 $Conf_0$ ... X_{16} Y_{16} $Conf_{16}$
Frames
Shape (n_samples,time_step,51)
Classification Model
CHEATING
NON-CHEATING

Figure 6. The architecture of a cheating detection behavioral model

The input for the models is transformed into a 3-dimensional matrix of shape (n_samples, time_step, 51) for the LSTM network. For the machine learning models, the input is converted into a 2-dimensional matrix by combining n_samples and time_step, resulting in a matrix of shape (n_samples*time_step, 51). Finally, when obtaining the training parameters of the model, we repeat the aforementioned steps, and the output consists of bounding boxes for each object marked as either cheating or non-cheating.

3.1. Skeletons use YOLO-Pose

There are several methods used for recognizing the poses of objects in videos, such as YOLO-Pose [16], OpenPose [21] and BlazePose [22]. In this study, we have chosen the YOLO-Pose model to recognize objects and assign skeletal keypoints. YOLO-Pose was chosen because the model is capable of multi-person 2D pose estimation and object detection. It can be run on both CPU and GPU devices, and it is easy to install on a computer. Among the pose estimation models, there are two methods: top-down and bottom-up. The top-down method, also known as the two-stage method, starts by detecting the positions in the image and then determining the body keypoints of the objects. Some models applied in this method include BlazePose [22]and AlphaPose [23]. The top-down method provides a fast solution but may not guarantee the accuracy of the model. It relies on the results of object detection, and the runtime of the top-down method is proportional to the number of people being recognized, which can be time-consuming [24], [25]. Models like OpenPose and ActionVLAD [26] belong to the bottom-up method, where all keypoints on the body of the object are detected, and the relationships between keypoints are used to determine the orientation and position of the body. This method achieves high accuracy but requires processing a large number of keypoints simultaneously, leading to increased computational time, and it can also result in misidentification when the input is not clear [27].

In the YOLO-Pose model, the approach is similar to the bottom-up method and is based on the YOLOv5 [28]. The input image is passed through the darknet backbone, which generates feature maps. The path aggregation network (PANet) is then used to aggregate feature maps from multiple scales. The output of PANet is then processed by the head to perform detection [16]. Finally, the model outputs the parameters of the bounding box and keypoints. Besides, we also compare YOLO-Pose, OpenPose, and BlazePose in Table 1.

Table 1. Compare YOLO-Pose, OpenPose, and BlazePose

Model	GPU and CPU support	# of keypoints	AP50 (COCO)	# of persons
BlazePose [22]	Only CPU	33	-	Single-person
OpenPose [21]	Only GPU	135	84.9	Multi-person
YOLO-Pose [16]	CPU and GPU	18	90.3	Multi-person

3.2. Object tracking

In YOLO-Pose, object detection can be performed using bounding boxes on images or videos. It returns an array containing the coordinates, width, height of the bounding boxes, and the confidence score of the detected objects. However, object detection is not always consistent in the order of detection across consecutive frames due to environmental influences. The order of detection for multiple objects in YOLO-Pose may have discrepancies. In behavior recognition, high accuracy is required, and there should be no overlap between the actions of one object and another. To solve this issue, we apply object tracking using the IoU algorithm [29]. The object tracking algorithm aims to solve the problem of bounding boxes transitioning between frames, ensuring that the objects are detected in the correct order in consecutive frames. This helps to obtain more accurate sequences of keypoints for each object, without mixing the parameters of different objects. The IoU algorithm as shown in (1) is a commonly used method for object tracking.

$$\text{IoU} = \frac{Box(a) \cap Box(b)}{Box(a) \cup Box(b)} \quad (1)$$

It is used to track objects in consecutive video sequences by comparing the overlapping regions of consecutive frames. The IoU value is calculated by dividing the area of overlap by the area of the union of the two bounding boxes. Specifically, the bounding box of the object in the first frame, t, is compared with the next bounding box in frame t+1. To calculate the intersection area between two boxes (a, b), the width of box a is: $W_a = X_{a\_max} - X_{a\_min}$, the height is: $H_a = Y_{a\_max} - Y_{a\_min}$, the width of box b is: $W_b = X_{b\_max} - X_{b\_min}$, and the height is: $H_b = Y_{b\_max} - Y_{b\_min}$. The intersection area of the two boxes is calculated as follows: $(W_a + W_b - (X_{b\_max} - X_{a\_min})) * (H_a + H_b - (Y_{b\_max} - Y_{a\_min}))$.

In our experiments, we set IoU to 0.5, it is considered that the bounding boxes belong to the same object in two consecutive frames. This method was chosen because the objects in the testing environment generally do not move significantly but rather remain stationary to perform actions. Calculating IoU is relatively fast, making it suitable for real-time processing, and the performance of the IoU tracker depends on the speed and accuracy of the object detection.

3.3. Ac-LSTM model

Hochreiter and Schmidhuber [30] suggested LSTM is a special type of recurrent neural network (RNN). It can learn long-term relationships between elements in a data sequence. Therefore, LSTM is a model that is applied to process sequential data such as time series prediction, speech recognition, and music composition. In this problem, it is used to recognize sequences of actions that occur sequentially in consecutive image frames.

In the problem of cheating behavior classification, we established relationships between input parameters (i.e., skeletal frame parameters) to achieve better classification performance. Here is a more detailed introduction to the structure and principles of LSTM model cells. Two functions, sigmoid and tanh, are mainly used in LSTM cells. The structure of an LSTM cell from left to right includes three gates: the forget gate, input gate, and output gate, as shown in Figure 7.

The forget gate shown in (2) controls the extent to which the previous memory state h_{t-1} should be forgotten based on the current input x_t. The input gate, shown in (3), controls the extent to which the new input should be written into the memory. Finally, the output gate, shown in (4), controls the extent to which the current memory state should be the output h_t.

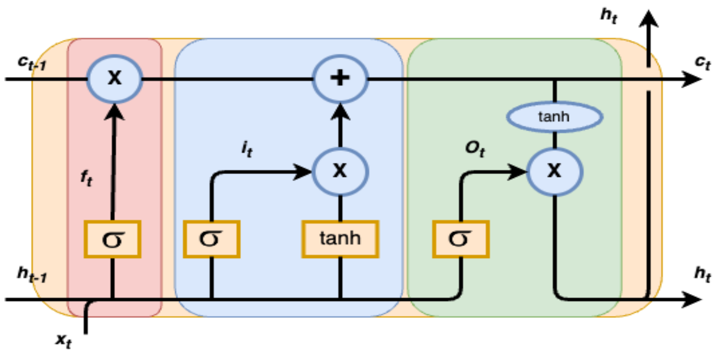


Figure 7. The architecture of LSTM model

$$f_t = \sigma(W_f[h_{t-1}, x_t] + b_f) \tag{2}$$

$$i_t = \sigma(W_i[h_{t-1}, x_t] + b_i) \tag{3}$$

$$o_t = \sigma(W_o[h_{t-1}, x_t] + b_o) \tag{4}$$

From the original LSTM model, we propose the Ac-LSTM model for cheating behavior detection. Figure 8 shows the multi-layer model, consisting of 4 LSTM layers with 50 units in each layer. The model takes as input a time series of the skeleton frames of an object. First, we need to convert the input into a vector of the form X (n_samples, time_steps, n_features). Here, n_samples is the number of time_steps layers inputted into the model for training, time_steps is the number of skeleton frames inputted, and n_features is the number of features used to describe each frame in the video. In this problem, we only use 51 features for behavior classification. Next, the model is passed through 4 LSTM layers, each of which has a dropout layer with a rate of 0.2 added. However, using multiple layers can also lead to other issues, such as slow learning speed and increased computational complexity. Therefore, selecting an appropriate number of layers should be based on the specific characteristics of the problem. Applying dropout allows the model to learn features instead of memorizing the training data excessively, which can lead to overfitting. Finally, the output of the last LSTM layer is passed to the dense layer with a Sigmoid activation function to predict the output value. Adam is used to update the weights and adjust the learning_rate, and the loss function used is binary cross-entropy, as described in (5). The specific parameter settings of the Ac-LSTM are shown in Table 2.

$$BCE = -\frac{1}{N}\sum_{i=1}^{n}(y_i \cdot log(\hat{y}_i) + (1 - y_i) \cdot log(1 - \hat{y}_i)) \tag{5}$$

Table 2. Parameter setting of Ac-LSTM model

Name	Parameter
Learning rate	0.001
Input dimension	10×51
Output category	2
Batch	32
Dropout	0.2
Hidden node number	50
LSTM layer number	4

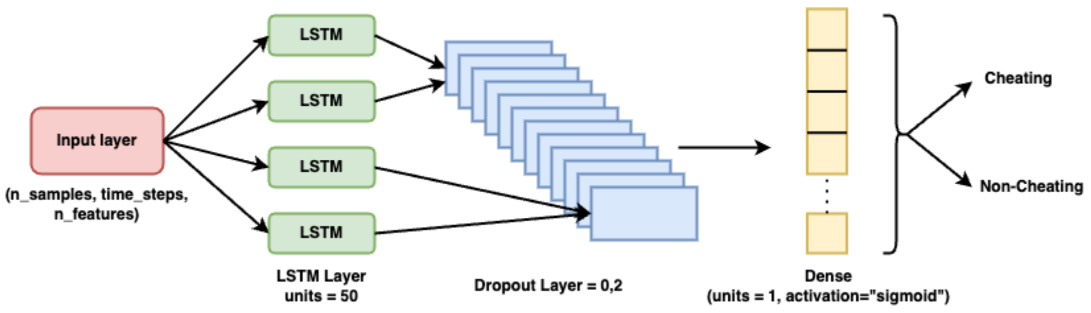


Figure 8. Ac-LSTM cheating classification model

3.4. Evaluation metrics

The model's performance is assessed using different measurements such as accuracy, precision, recall, and F1-score. In this evaluation, we have two categories: cheating and non-cheating, which are considered positive and negative, respectively. From there, we obtain the following classifications: true negatives (TN) refer to samples in the results that are correctly classified as not belonging to the required class. False negatives (FN) are samples mistakenly classified as not belonging to the required class, even though they do belong to the class. True positives (TP) are samples in the data that are correctly categorized as belonging to the desired class. False positives (FP) are samples in the data that are incorrectly categorized as belonging to the target class, even though they do not belong to the class.

Accuracy is evaluated as the ratio of the number of correct predictions (TP+TN) to the total number of data points in the test set (TP+TN+FN+FP). For classification problems with highly imbalanced class distributions, a commonly used metric is precision-recall. Precision is when we predicted a "cheating", how often is that correct and defined as the ratio of the number of true positives (TP) to the number of positive predictions (TP+FP). Recall is when it's a "cheating" behavior, how often do we predict "cheating". In other words, how good is our classifier at identifying all occurrences of our class and defined as the ratio of true positives (TP) to the total number of actual positives (TP+FN). F1-score is the harmonic mean (punishes extreme values) of both precision and recall. It's more informative than accuracy alone as it. The formulas for precision, recall, and F1-score are described in (6)-(8), respectively.

$$\text{Precison} = \frac{TP}{TP+FP} \tag{6}$$

$$\text{Recall} = \frac{TP}{TP+FN} \tag{7}$$

$$\text{F1} - \text{score} = \frac{2(Precision \times Recall)}{Precision+Recall} \tag{8}$$

4. RESULT AND DISCUSSION

We conducted experiments on colab, a Jupyter notebook environment developed by Google Research. It provided access to Tesla T4 GPUs for the free version. When running the cheating detection model on the colab GPU environment, the average frames per second (FPS) achieved in the videos was around 10-12 FPS. The input data was saved in a .csv file with 10,277 rows of cheating samples and 35,258 rows of non-cheating samples. The data was split into 80% for training and 20% for testing to evaluate the model. For LSTM models, the input is a 3-dimensional array consisting of n_samples, time_steps and n_features. As for statistical models such as SVM, DT, RF, and XGBoost, the input is flattened into a 2-dimensional array to fit those models.

From the results in Table 3 and Figure 9, we can see that the model performance evaluation metrics are very promising. Among them, the XGBoost model achieved the best results with 95%, 95%, 90%, and 92% on the evaluation metrics of accuracy, precision, recall, and F1-score, respectively. The Ac-LSTM model had the second-highest performance, while the DT model had the lowest results, with 89%, 85%, 82%, and 83% on accuracy, precision, recall, and F1-score, respectively.

We conducted experiments on the trained model. The integration of IoU after applying YOLO-pose to detect skeletons aims to precisely determine the skeletal framework of each object in sequential image frames within a video. This facilitates favorable training input for the Ac-LSTM model, where the input data consists of a sequence of object skeletons. When passing the data through the model, the output consists of bounding boxes drawn around the objects and the assignment of skeletal structures to each object, along with the classification of cheating and non-cheating behaviors labeled on the frames, as shown in Figure 10.

Table 3. Comparison of precision, recall, and F1-score using different algorithms

Algorithm	Accuracy	Precision	Recall	F1-Score
SVM	0.91	0.91	0.81	0.85
DT	0.89	0.85	0.82	0.83
RF	0.92	0.94	0.82	0.87
XGBoost	0.95	0.95	0.90	0.92
Ac-LSTM	0.94	0.94	0.88	0.90

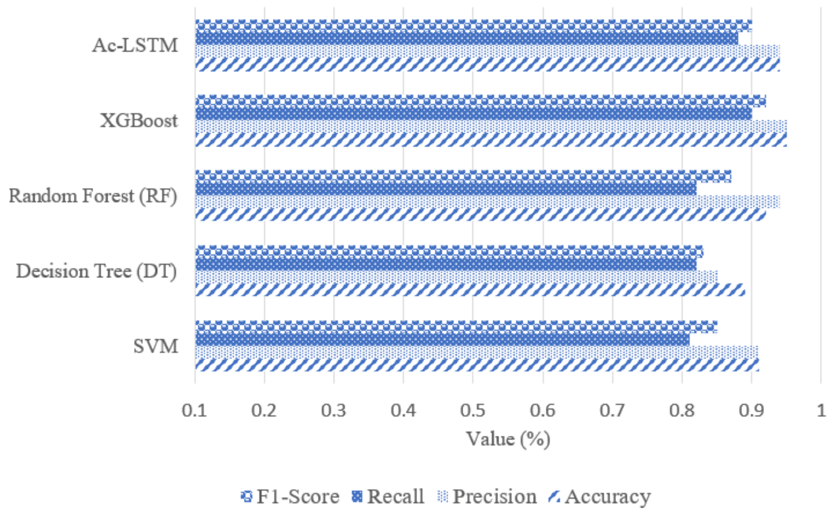


Figure 9. The statistics of the results of models on various metrics

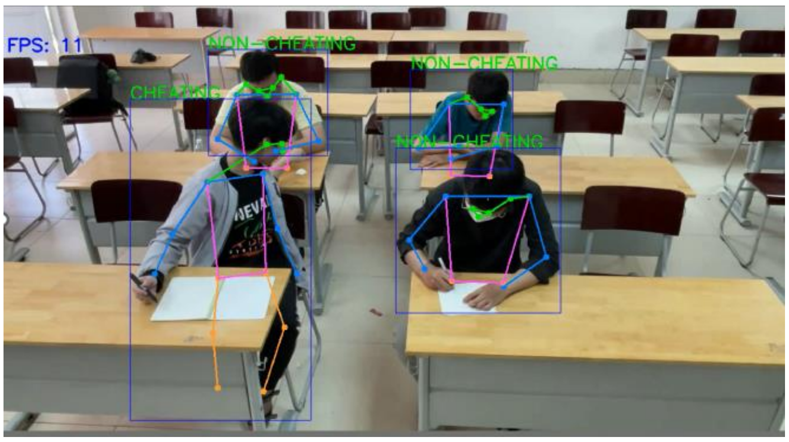


Figure 10. Detecting cheating behaviors in videos

In this study, there are limitations, such as not removing non-essential keypoints. As the behaviors of most students in the classroom revolve around sitting postures, the surveillance camera primarily focuses on the upper body area, and the lower body region is mostly occluded. Therefore, the keypoints of the lower body are considered less significant. These keypoints can be removed to reduce the weight and noise of the classification model. In addition, the self-built dataset for the model also encountered many issues in some cases, such as when students “ponder” and look indifferently to the left or right, similar to “copying answers”, which sometimes leads to misjudgment by the model. We will further investigate and address these challenges in our future research.

5. CONCLUSION

In this research, we have developed a support model for detecting cheating in exams by applying machine learning models to classify cheating or non-cheating behaviors on self-collected datasets. The experimental results show that the system achieved high accuracy and is capable of detecting various types of cheating behaviors, providing active support in monitoring and detecting cheating. Besides, a dataset has been constructed to serve the problem of detecting cheating through pose estimation in videos. In addition, we have proposed an Ac-LSTM network architecture for detecting cheating based on posture. Furthermore, we have also analyzed and evaluated the factors affecting the accuracy of the system and proposed various algorithms to identify students’ abnormal behaviors through the motion of skeletons, as well as some improvements to enhance the system’s effectiveness. Based on the experimental results, we believe that the cheating detection system we proposed can meet the requirements of educational org5anizations and help reduce cheating behavior in exams. In the future, the model will continue to be researched and developed to enhance its accuracy and practicality. Specifically, there will be a focus on expanding the dataset and incorporating the graph convolutional networks (GCN) model to represent skeleton data.

REFERENCES

[1] M. A. M. Quintos, “A study on the prevalence and correlated of academic dishonesty in four undergraduate degree programs,” *Asia Pacific Journal of Multidisciplinary Research*, vol. 5, no. 1, pp. 135–154, 2017.

[2] C. G. King, R. W. Guyette, and C. Piotrowski, “Online exams and cheating: An empirical analysis of business students’ views,” *Journal of Educators Online*, vol. 6, no. 1, Jan. 2009, doi: 10.9743/JEO.2009.1.5.

[3] D. Rigby, M. Burton, K. Balcombe, I. Bateman, and A. Mulatu, “Contract cheating and the market in essays,” *Journal of Economic Behavior and Organization*, vol. 111, pp. 23–37, Mar. 2015, doi: 10.1016/j.jebo.2014.12.019.

[4] F. Hussein, A. Al-Ahmad, S. El-Salhi, E. Alshdaifat, and M. Al-Hami, “Advances in contextual action recognition: automatic cheating detection using machine learning techniques,” *Data*, vol. 7, no. 9, p. 122, Aug. 2022, doi: 10.3390/data7090122.

[5] Y. Xing and J. Zhu, “Deep learning-based action recognition with 3D skeleton: a survey,” *CAAI Transactions on Intelligence Technology*, vol. 6, no. 1, pp. 80–92, Mar. 2021, doi: 10.1049/cit2.12014.

[6] J. Nishchal, S. Reddy, and P. N. Navya, “Automated cheating detection in exams using posture and emotion analysis,” in *2020 IEEE International Conference on Electronics, Computing and Communication Technologies (CONECCT)*, Jul. 2020, pp. 1–6, doi: 10.1109/CONECCT50063.2020.9198691.

[7] N. Soman, M. N. R. Devi, and G. Srinivasa, “Detection of anomalous behavior in an examination hall towards automated proctoring,” in *Proceedings of the 2017 2nd IEEE International Conference on Electrical, Computer and Communication Technologies, ICECCT 2017*, Feb. 2017, pp. 1–6, doi: 10.1109/ICECCT.2017.8117908.

[8] N. K. Sharma, D. K. Gautam, S. Rathore, and M. R. Khan, “CNN implementation for detect cheating in online exams during COVID-19 pandemic: a CVRU perspective,” *Materials Today: Proceedings*, May 2022, doi: 10.1016/j.matpr.2021.05.490.

[9] A. Arinaldi and M. I. Fanany, “Cheating video description based on sequences of gestures,” in *2017 5th International Conference on Information and Communication Technology, ICoIC7 2017*, May 2017, pp. 1–6, doi: 10.1109/ICoICT.2017.8074679.

[10] Z. Li, Z. Zhu, and T. Yang, “A multi-index examination cheating detection method based on neural network,” in *Proceedings - International Conference on Tools with Artificial Intelligence, ICTAI*, Nov. 2019, vol. 2019-November, pp. 575–581, doi: 10.1109/ICTAI.2019.00086.

[11] G. E. V. Bancud and E. V. Palconit, “Human pose estimation using machine learning for cheating human pose estimation using machine learning for cheating detection,” *Researchgate*, no. April, 2021.

[12] M. Malhotra and I. Chhabra, “Automatic invigilation using computer vision,” in *Proceedings of the 3rd International Conference on Integrated Intelligent Computing Communication & Security (ICIIC 2021)*, 2021, vol. 4, doi: 10.2991/ahis.k.210913.017.

[13] J. Redmon and A. Farhadi, “YOLOv3: An Incremental Improvement,” 2018, *arXiv:1804.02767*.

[14] X. Zhang, X. Zhou, M. Lin, and J. Sun, “ShuffleNet: an extremely efficient convolutional neural network for mobile devices,” in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Jun. 2018, pp. 6848–6856, doi: 10.1109/CVPR.2018.00716.

[15] V. A. Sangalli, G. Martinez-Munoz, and E. P. Canabate, “Identifying cheating users in online courses,” in *IEEE Global Engineering Education Conference, EDUCON*, Apr. 2020, vol. 2020-April, pp. 1168–1175, doi: 10.1109/EDUCON45650.2020.9125252.

[16] D. Maji, S. Nagori, M. Mathew, and D. Poddar, “YOLO-pose: enhancing YOLO for multi person pose estimation using object keypoint similarity loss,” in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, Jun. 2022, vol. 2022-June, pp. 2636–2645, doi: 10.1109/CVPRW56347.2022.00297.

[17] B. Schölkopf, “SVMs - a practical consequence of learning theory,” *IEEE Intelligent Systems and Their Applications*, vol. 13, no. 4, pp. 18–21, Jul. 1998, doi: 10.1109/5254.708428.

[18] L. Rokach and O. Maimon, “Decision trees,” in *Data Mining and Knowledge Discovery Handbook*, New York: Springer-Verlag, 2006, pp. 165–192.

[19] Y. L. Pavlov, "Random forests," in *Random Forests*, New York, NY: Springer New York, 2019, pp. 1–122.
[20] T. Chen and C. Guestrin, "XGBoost: A scalable tree boosting system," in *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Aug. 2016, vol. 13-17-Augu, pp. 785–794, doi: 10.1145/2939672.2939785.
[21] Z. Cao, T. Simon, S. E. Wei, and Y. Sheikh, "Realtime multi-person 2D pose estimation using part affinity fields," in *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, Jul. 2017, vol. 2017-January, pp. 1302–1310, doi: 10.1109/CVPR.2017.143.
[22] V. Bazarevsky, I. Grishchenko, K. Raveendran, T. Zhu, F. Zhang, and M. Grundmann, "BlazePose: on-device real-time body pose tracking," 2020, *arXiv:2006.10204*.
[23] H. S. Fang, S. Xie, Y. W. Tai, and C. Lu, "RMPE: Regional multi-person pose estimation," in *Proceedings of the IEEE International Conference on Computer Vision*, Oct. 2017, vol. 2017-October, pp. 2353–2362, doi: 10.1109/ICCV.2017.256.
[24] M. Li, Z. Zhou, J. Li, and X. Liu, "Bottom-up pose estimation of multiple person with bounding box constraint," in *Proceedings - International Conference on Pattern Recognition*, Aug. 2018, vol. 2018-August, pp. 115–120, doi: 10.1109/ICPR.2018.8546194.
[25] J. Li, W. Su, and Z. Wang, "Simple pose: rethinking and improving a bottom-up approach for multi-person pose estimation," *AAAI 2020 - 34th AAAI Conference on Artificial Intelligence*, vol. 34, no. 07, pp. 11354–11361, Apr. 2020, doi: 10.1609/aaai.v34i07.6797.
[26] R. Girdhar, D. Ramanan, A. Gupta, J. Sivic, and B. Russell, "ActionVLAD: learning spatio-temporal aggregation for action classification," in *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, Jul. 2017, vol. 2017-January, pp. 3165–3174, doi: 10.1109/CVPR.2017.337.
[27] M. Kresović and T. D. Nguyen, "Bottom-up approaches for multi-person pose estimation and it's applications: a brief review," 2021, *arXiv:2112.11834*.
[28] G. Jocher *et al.*, "YOLOv5," *Ultralytics*, 2021. https://github.com/ultralytics/yolov5 (accessed May 11, 2023).
[29] E. Bochinski, V. Eiselein, and T. Sikora, "High-Speed tracking-by-detection without using image information," in *2017 14th IEEE International Conference on Advanced Video and Signal Based Surveillance, AVSS 2017*, Aug. 2017, pp. 1–6, doi: 10.1109/AVSS.2017.8078516.
[30] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, Nov. 1997, doi: 10.1162/neco.1997.9.8.1735.

BIOGRAPHIES OF AUTHORS

Nha Tran received BS. Computer Science (2014), MS. Computer Science (2019) from HCM University of Education. He is currently a lecturer of Information Technology Faculty, Ho Chi Minh city University of Education. His research interest includes computer vision, information retrieval, affective computing, machine learning. He was recipient of the Best Paper Award from ICAEIC 2020. You can contact him via email: nhatt@hcmue.edu.vn.

Minh Nguyen He is currently a 3rd-year student at the Ho Chi Minh City University of Education, Ho Chi Minh City, Vietnam. His research includes machine learning, deep learning, and computer vision. He can be contacted at email: nguyendatminhvn@gmail.com.

Tri Le He is currently a 3rd-year student at the Ho Chi Minh City University of Education, Ho Chi Minh City, Vietnam. His research includes machine learning, deep learning, and computer vision. He can be contacted at email: trilecs6102@gmail.com.

Tuong Huynh He is currently a 2nd-year student at Ho Chi Minh University of Education, Ho Chi Minh City, Vietnam. His research includes machine learning, deep learning, and computer vision. He can be contacted at email: manhtuonh123@gmail.com.

Tinh Nguyen Tinh Nguyen received the B.S. degree in Computer Science from the Ho Chi Minh University of Education (HCMUE), Vietnam, in 2022. Since September 2022, he has been working as a research assistant at the Smart Structure Laboratory. He is currently pursuing an M.S. degree at Chungbuk National University, South Korea. His research interests include machine learning, deep learning and its applications to education, and structural health monitoring. He can be contacted at email: xngtinh@gmail.com.

Tuan Nguyen received his B.Sc. (Computer Science) from Ho Chi Minh City University of Education in 2020. He is currently a Software engineer at Information Technology Department, Ho Chi Minh City University of Education. His research includes computer vision, intelligent system, and Data analysis. You can contact him via email: tuannv@hcmue.edu.vn.