INTERNSHIP REPORT

Title: An Integrated Approach for Visualizing Student Activity During Distance Education

Period:	06/06/2022 – 29/07/2022
Host MRG:	BIO-SCENT
Supervisor:	Prof. Andreas Lanitis
Intern:	Michalis Kontos
Main theme:	Computer vision, Human activity recognition

1 Introduction

During the past years, distance education has been gaining popularity and this has been accelerated by the covid-19 pandemic. Many people have been working from home and students have been attending classes online. This caused privacy concerns for many people and especially for young students and their parents. This is a big concern, and for this reason in many countries the use of the webcam during teleducation is forbidden. When no camera is used during distance learning, the teacher has no optical contact with the students and as a result students tend to lose their concentration. The aim of our work is to provide a system that will allow teachers to get informed about student activities during these online classes but without having access to video input from them. In the proposed system, images are captured from the students' webcam and are processed locally on their machine in order to identify their actions. This information only is then sent to the teacher which acts as the server side of the application. The Server side contains a 3D visualization of the classroom where each student is represented by an avatar, animated to reflect student actions. This method will keep the teacher informed of the students' actions without violating any privacy barriers. The proposed architecture of the system is shown below in figure 1, where multiple students are able to connect and most importantly, their information is processed locally.

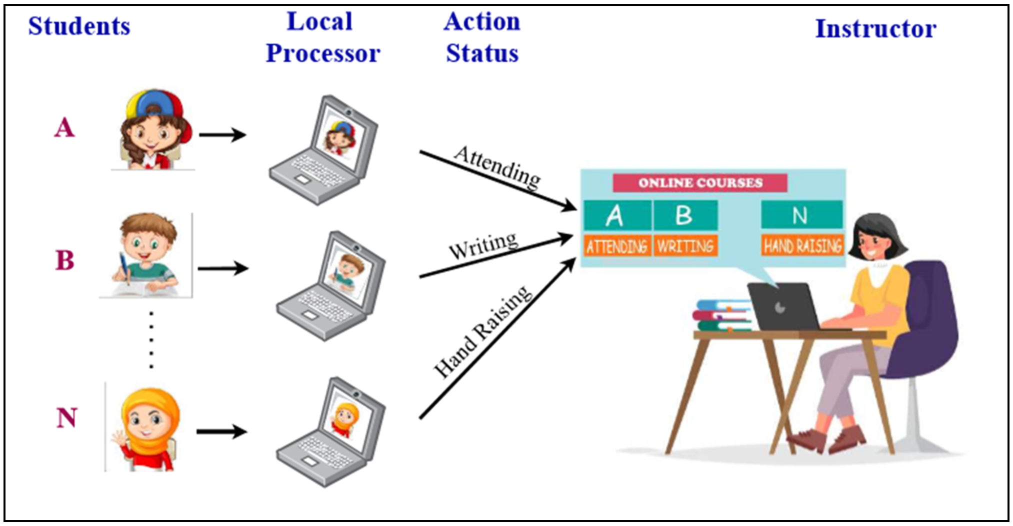

Figure 1: System Overview

In the next sections we present a literature review on the topics of human activity recognition, expression recognition, attendance detection and student monitoring techniques and then present the methodology adopted in our system. Next, the results are outlined along with future work to be done.

2 Subject and Objectives

My work for this internship focused on creating a user application to be integrated in the work carried out by Demitriadou [17] who developed the machine learning model along with the virtual classroom. The user application to be developed aims to be used on the students' side of the system and must be able to communicate with the teacher's side. This application must be able to classify the user's actions, taking input from the webcam. To do this, it must include the machine learning model developed in [17].

My learning goals for this internship is to research and study how machine learning models are built along with different techniques and architectures. This also mainly included work done on Human Activity Recognition. Then is to successfully implement the task assigned to me while listening for suggestions and recommendation from more experienced people. A lot of research was done in order to complete the application, including communication protocols in Unity and python and how machine learning models are inserted into other applications. Some of the earlier implementations did not work but eventually the final design described below was achieved.

3 Literature review

Human activity recognition became a popular challenge in Computer Vision in the past years. Many contributions have been made with different people proposing new ways of analyzing multiple activities.

Franco et al. [1] use HAR algorithms that can be used for smart home integration in assisting with unusual behavior recognition such as break-ins, dangerous situations or even healthcare. In addition to image capturing as data input, Franco et al [1] also uses a Kinect sensor in parallel with RGB images. This gives the algorithm additional data such as skeleton joint operations for supplementary accuracy. Their activity recognition approach is based on hand-crafted features instead of using deep learning and neural networks as many others have used in other studies. As video streaming is fed into their system, where each frame is then represented by a set of angles derived from the human skeleton hence summarizing the position of various body parts. Two classifiers were trained using features extracted from skeleton and RGB images. One classifier uses Random Forest from skeleton images and the other consists of a set of Support Vector Machines. Their outputs are combined using decision-level fusion with equal weights between the classifiers.

Hirooka et al. [2] distinguish between the two different types of input data for HAR, these being sensor-based or vision-based, where the latter can be subcategorized as image or video based. In this instant, this paper discusses classification from still images using four convolution neural networks to make a feature fusion-based ensemble. In each of the branches, an attention module is used to extract contextual information from a feature map into a more discriminative one. After their use, the addition of fully connected layers along with dense layers is investigated. Transfer learning was also adopted for the four CNNs, using pre-trained models. The benefits of using the transfer learning technique are also discussed such as the efficiency of finding an effective hypothesis by the transfer of information learned in another activity. The utilization of pre-trained models shortens the time for learning without the need for huge amounts of data, making it ideal for HAR.

Snehitha et al. [3] also make use of still images, with more complicated activities being broken down into smaller components. Instead of using just static images frames,   a video-based approach benefits by the use of spatio-temporal features. Again, the use of deep learning techniques is adopted in this work, using CNNs with shared parameters. The breakdown of CNNs is referred to and how the convolution part employs many filters capable of extracting the features needed from data while preserving the spatial information. The link between HAR with posture estimation and scene interpretation is made and how techniques from one can be applied to the other. Snehitha et al. [3] claim that deep learning is more suitable for this application rather than others as seen in the first paper discussed. The steps followed are standard for most neural networks and includes the pre-processing of data to eliminate irrelevant information, data augmentation for increased amount of training data, feature extraction to distinguish characteristics of faces, feature collection to assign ranks to features and finally action classification.

Jaouedi et al. [4], HAR is performed on video data rather than individual images, using Recurrent neural networks (RNN) for activity classification and CNN for feature extraction. The video being from a static camera means that the background is static as well hence techniques like Gaussian background, kernel density estimation and visual background extraction can be used. The two major stages in this system are again the feature extraction and action classification. Features being visual like pixel intensity and texture but also temporal like motion direction or trajectory path. The use of video in this case gives additional information unlike still images. The models can exploit these different types of features using the Kalman filter and the Inception V3 model.

Moving on to another important aspect of our system, expression recognition is vital for the understanding of students' behavior. As in our case, Lu et al [5] examine the lack of interaction between teachers and students during online learning. The understanding of students' emotions is the solution to such problems. A very important distinction is being made here, the difference between academic and basic emotions of students. Given as a basic example that frowning in the classroom can mean academic confusion whereas frowning outside the classroom can mean something different. Expression recognition covers a variety of disciplines including computer vision, machine learning and behavioral science. Once more the use of CNN algorithms is made along with data augmentation. The VGG15 CNN architecture is used in this work..

Li et al. [6] compare the deep learning methods with two other feature-based approaches as a baseline, these being LBP&SVM and SIFT&SVM. In this case the focus of the work is on assisting visually impaired people to perceive emotions. The same problems are addressed as above (insufficient dataset for training etc). The deep learning method shows superiority on non-constrained images, but feature-based approaches are better on well-posed faces.

Another use of expression recognition is summarized in Fathallah et al, [7] where it is used in human-machine interaction. Their facial expression recognition system comprises three stages, face acquisition, facial feature extraction and the classifier construction. However, by taking advantage of a CNN, the feature extraction and the classifier can be merged, highlighting another benefit of using deep learning. The need of creating a new, larger database is discussed and once more the use of VGG architecture is undertaken. Repeated training of the model architecture is done for improved accuracy. Their proposed network includes four convolutional layers with three max-pooling layers followed by a fully connected layer and the SoftMax output layer for six expression classes.

Another aspect of our project is the attendance detection, since apart from the expression recognition, teachers may wish to only know if students pay attention or not. Varadharajan et al [8] implement an automatic attendance management system where a fixed camera is placed in a classroom to detect faces. This is a different application to ours but the use of face detection is vital. This is done by background subtraction from the images obtained and using eigen face algorithms as a set of eigenvectors. Hartanto et al [9] describe a similar system using a different set of algorithms. The face detection in this case is done using skin color detection and the Haar Cascade algorithm. An alignment process follows, containing face features and a normalization process. Then a feature extraction process and classification using the LBPH algorithm (Local Binary Patterns Histograms).

This project aims to extend the work described by Fuzail et al [10] where the authors address the issue of students losing focus in class while the teacher has no vision of them and proposes a solution of an AI system to examine the students' actions. A comparison between different architectures of deep learning have been tested for the problem of student action recognition. The architectures include the faster R-CNN, SqueezeNet, GoogleNet and the Inception-v3. Transfer learning is used with a starting point of a pre-trained model from ImageNet and network weights of certain layers being adapted. The adam optimizer is used to minimize cross-entropy loss function. The final model had to be computationally inexpensive since it would be used on students' devices or even mobile devices. The best approach based on the experimental results is the SqueezeNet network allowing real-time operation and a reasonable accuracy.

Student monitoring must be done to obtain data required and their analysis is vital to the aforementioned instances. Shah et al [11] propose a system to provide physical and emotional analysis of students to their teacher. Eye and head pose tracking is implemented along with emotions displayed using expressions. A final classification of attentiveness or not is done. This approach needs multiple machine learning models to be present for the analysis of various behavioral components and to then be integrated into a final one for the determination of level of attentiveness. The proposed system makes use of video frames taken per minute and maps 68 points of facial structure. It detects changes in these points on eyelids, retinas, lips or eyebrows. Then images pass through multiple detectors such as drowsiness, emotion and head-pose before making the final classification.

Chowdhury et al [12] propose a biometric AI system to detect faces and recognize them in real-time. The aim is to detect faces from video streams as well and recognize them by cross referencing them with a database. The proposed system has the capacity to detect multiple people in a single frame hence its use is for attendance taken during in-classroom teaching (instead of signing sheets). The recognition is again done by a CNN model which was trained on three different face images where two were of the same person. It can extract 128 facial measurements and to tweak weights to make the vectors of the two same person images closer.

A more simple, less computationally expensive approach was taken in Mery et al, [13] since it makes use of a smartphone to capture images. It evaluates different face recognition algorithms and consists of different stages to achieve this such as Euclidean distance and cosine similarity between face vectors.

Thomas et al [14] also studied student engagement as facial expressions, head pose, and eye gaze were also analyzed with two classification outcomes of engaged or distracted. To get individual faces, a combined matching and tracking framework was used, which essentially is an object tracking framework (face being the object in this case). A camera is positioned directly in front of the students imitating a teacher, making it easier to identify if students looked towards it or are distracted (looking to the left, right or up). Support Vector Machine algorithms (SVM) are examined which are a type of supervised learning using both linear and non-linear classification in comparison with Logistic Regression which estimates association between categorical dependent variables and various independent ones using logistic functions.

4 Methodology

4.1 System Overview

Our system includes two sides, the client and server side as shown in figure 2. Each student acts as a client and connects to the teacher's side, acting as the server. Both the client and the server are python scripts and are communicating using python sockets [15]. On the teacher's side, the python program is also connected to the Unity application developed which includes the 3D visualization of the classroom.

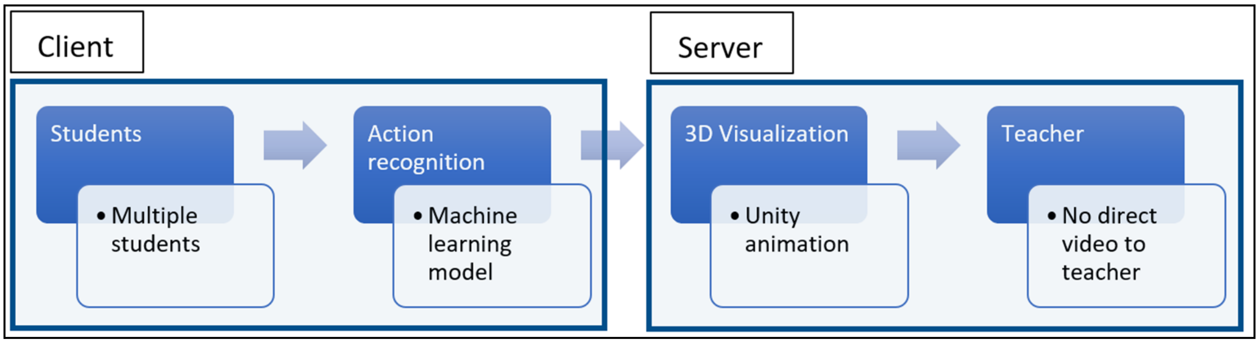

Figure 2: System architecture

4.2 Client

The application that the students' will have on their devices incorporates a machine learning model previously developed by Dimitriadou [17]. The model is based on the GoogleNet architecture and includes 27 layers including the pooling layers. It can manage an accuracy of 94.32% on previously unseen images. It is trained to identify 7 major activities that the students perform including absent, attending, hand raising, looking elsewhere, telephone call, using phone and writing. Examples of students performing these actions can be seen in figure 3. The GoogleNet model was chosen for this task as it ensures the computational requirements of the students' devices are minimal for the classification process along with the significant reduction in error rate as compared to other architectures.

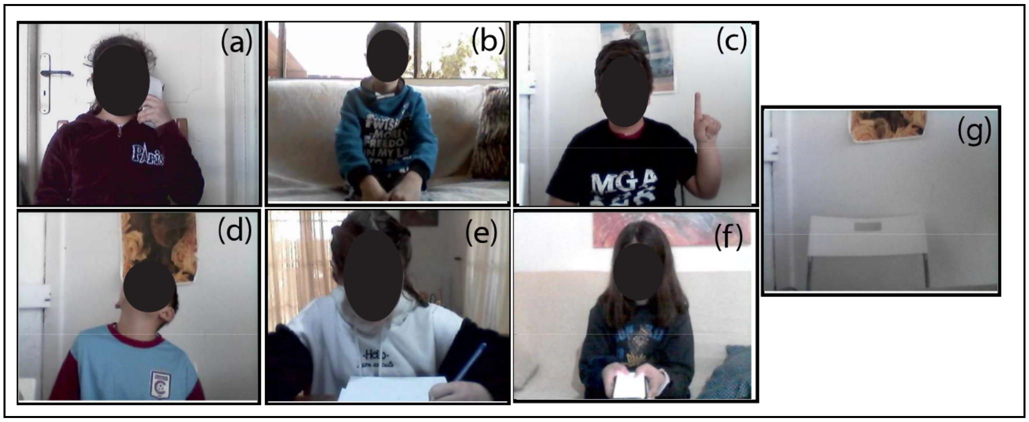

Figure 3: Students' actions (a) Telephone call (b) attending (c) hand raising (d) looking elsewhere (e) writing (f) using phone (g) absent

The client side is constantly performing classifications based on images captured through the webcam. The captured image is preprocessed to match the dimensions and color of the model's input and then classification is performed. Communication between the client and server is always open but messages are sent in predetermined intervals. This is currently set to 10 seconds and during that period, a python dictionary is used to keep track of the activities performed. After that time, a majority voting system decides which action to output as the final one and sends that over to the server.

The client has the ability to be used as a stand-alone program, without the need of the server side to be running. This is implemented for testing purposes and for the targeted devices to review if the model can be loaded and run correctly.

Multiple clients can be connected to the server. In that case, each client must identify itself with a unique ID which will be later used to integrate its output to the specific avatar. This is also a way to keep track of which student has been connected in the classroom. A check has been implemented to disallow two clients with the same ID to be connected to the server. The most recent connection out of the two will be closed and that client should attempt to reconnect with the correct or different ID. In a real setting student ID's will be provided by the instructor, so that he/she can keep track of the actions performed by each student.

4.3 Server

Over on the teacher's side, the server starts by listening for incoming connections. The client and the server must be connected using the same port, which is hard-coded into the python program. As soon as the client starts, the server establishes a connection.

The first task of the server is to verify the client with the ID it will send, as described above. If the ID is unique then the connection can proceed. The server remains at idle until a message is received by a client. This is done every 10 seconds, as the most performed action is transmitted.

The server is responsible for logging the action received into a text file. It should be noted that the server script resides inside the Resources folder of the Unity application for easier communication between the two. A single number is being written into the log file, indicating the activity. The Unity and python programs must have the same order of activities since the two are used interchangeably. However, Unity indexing starts from 1 rather than 0 as in python hence, this is something to look out for. Each student ID gets its own log file created in the folder, which is the one used by the avatar. The ID must match the one inside the virtual classroom else it will not be able to connect. This can become useful, as in a real situation, a name or a University ID can be used to match each virtual student to the real one.

The server also has the potential to serve multiple clients at once. This is done using the python thread module [16]. It provides a way to create multiple threads for light weight processes. Each one of those threads share the controls and data space of the program. Every time that a new client connects to the server, a new thread starts while also keeping track of its identifier. Inside each thread then, the server keeps track of each action received by that particular client along with the corresponding ID. As mentioned before, if the ID is invalid, then the thread is closed and the connection is dropped. Synchronization between each client is not needed at this point since the transmission of information is done once every 10 seconds and each of the clients do not need to do that simultaneously. This is easier to implement and also allows for fewer errors down the line. If a larger scale system needs to be established, then synchronization can be done for every thread using locks, to prevent each client sending data at a particular time and forcing them to communicate only when the server is able to address their connection.

In the Unity part of the server, each avatar reads the corresponding log file. This is to receive the information about which action to animate. The python script is responsible for continuously updating that file and not to delay each change. The time between each animation change has a lower bound coming from the time the “absent” animation takes. This is at 5 seconds, hence the Unity program should only read from the log file once every 5 seconds or more. However, this is tricky to implement inside the Unity program, hence, it only reads the log file every time the current animation is completed. This could cause irregularities and delay between receiving the performed action and showing the action in the virtual classroom. The virtual classroom from the teacher’s point of view can be seen in figure 4, where the avatars are in the process of animation and their corresponding action is written underneath them. It should be noted that the green color indicates a positive action meaning that the student is performing an action relevant to the educational space such as attending or hand raising whereas red indicates a not attending action.

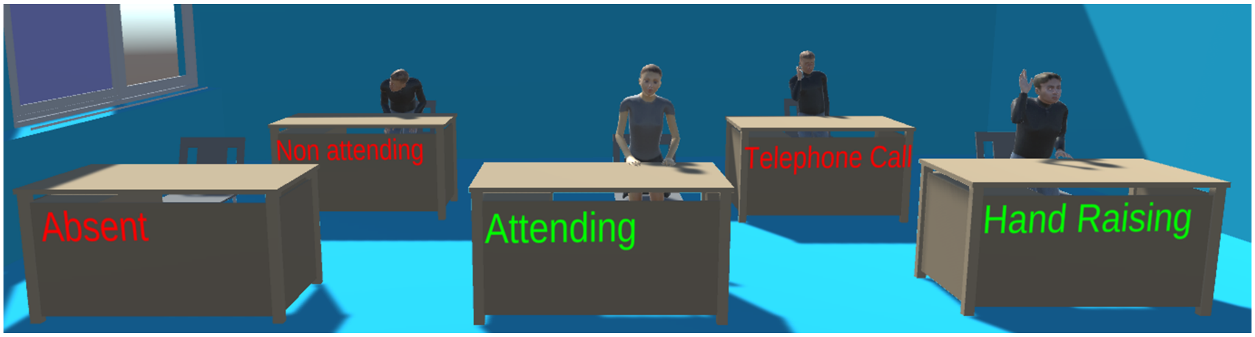

Figure 4: Virtual Classroom

4.4 Communication

Communication between the client and server side is done using python sockets. Sockets are a way of connecting two nodes on a network to communicate with each other. One of the nodes listens on a particular port at an IP, acting as the server side while the other node reaches out trying to establish a connection, acting as the client. The default protocol used in this specific implementation is the Transmission Control Protocol (TCP), which offers a reliable and in-order data delivery.

Messages are being encoded before transmitting using the UTF-8 encoding. This is the Unicode Transformation Format using 8-bit values hence every message needs to be encoded before sent otherwise it won’t be recognized by the server.

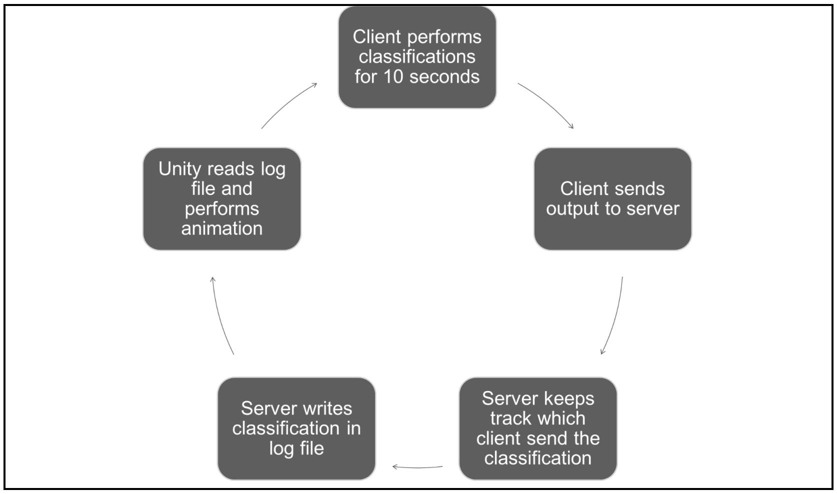
Client performs classifications for 10 seconds
Client sends output to server
Server keeps track which client send the classification
Server writes classification in log file
Unity reads log file and performs animation

Figure 5: System flow

The overall flow of application is summarized in figure 5. Starting off, the client begins classification, saving each outcome for 10 seconds. Then, using the majority voting system selects the one to send to the server. The server always keeps track of which client sent what and is able to save the message received in the correct log file. Finally, the Unity application reads the log file when every animation ends.

5 Results

The system implemented expresses promising results during the initial tests performed. It managed to work perfectly with the messages being transmitted instantly and the Unity animations executing the correct actions. The system was tested initially with 1 client on a single machine running both the client and the server side with no issues. The next step was to test the system with multiple clients and multiple machines. Due to the availability of computers, the most number of clients tested was 2, where 1 of them ran on the machine also running the server. The other client was connected from a different machine but still in the same network as the server machine. This test also ran smoothly without any issues present.

During my testing, the machine learning model performed exceptionally well, with classifications being mostly correct. In a few situations it could not specifically identify the intended action performed but that was due to actions being similar to one another, or images captured by the webcam did not fully portray the human in question.

What comes next is to test the system with even more machines to see how the server handles even more clients running concurrently and if any performance issues arise. Additionally, the clients to be tested would need to be situated in a different network, away from the server, but that would require additional configuration for the IP and ports to be found. Problems of port forwarding and firewalls blocking the incoming data may also occur.

The relevant files created during this internship can be found in the following link, along with a video demonstration.

https://github.com/MichalisKontos/CYENS-distance-education-app

6 Conclusion

In this study, the system developed is able to connect the teacher to their students without compromising their privacy. The use of computer vision assisted in the detection of actions performed by students and using network protocols, these information are being transmitted from the students to the teacher. A Unity application also helped in solving the problem of teachers not having optical contact with their students by allowing them to visualize their actions in a virtual space.

Early results show the promise of this work but further tests and implementations are required to achieve the optimal outcome. Hence, in the future we aim to improve the response time between server receiving results and being animated in Unity. Moreover, a more comprehensive evaluation needs to be carried out using teachers and students as the end users for meaningful feedback.

A more constructive and consequential model can also be implemented, using the already incorporated action recognition and another model for facial expression classification. This will allow the teacher to form a more complete picture of how the students view the material presented to them and allow them to recognize their understanding.

At a later stage, an online multi-user virtual space can also be created where teachers and students will be able to come together using Virtual Reality equipment.

7 Dissemination

The early results of the work described in this report were present at the 14th Cyprus Workshop on Signal Processing and Informatics, 2022. The extended abstract submitted to the workshop is shown in Appendix 1.

References

1. Franco, A., Magnani, A., & Maio, D. (2020). A multimodal approach for human activity recognition based on skeleton and RGB data. Pattern Recognition Letters, 131, 293–299. https://doi.org/ https://doi.org/10.1016/j.patrec.2020.01.010
2. K. Hirooka, M. A. M. Hasan, J. Shin and A. Y. Srizon, "Ensembled Transfer Learning Based Multichannel Attention Networks for Human Activity Recognition in Still Images," in *IEEE Access*, vol. 10, pp. 47051-47062, 2022, doi: 10.1109/ACCESS.2022.3171263.
3. B. Snehitha, R. S. Sreeya and V. M. Manikandan, "Human Activity Detection from Still Images using Deep Learning Techniques," *2021 International Conference on Control, Automation, Power and Signal Processing (CAPS)*, 2021, pp. 1-5, doi: 10.1109/CAPS52117.2021.9730709.
4. Jaouedi, Neziha, Francisco J. Perales, José M. Buades, Noureddine Boujnah, and Med S. Bouhlel. 2020. "Prediction of Human Activities Based on a New Structure of Skeleton Features and Deep Learning Model" *Sensors* 20, no. 17: 4944. https://doi.org/10.3390/s20174944
5. lu, W., Bian, C., Zhang, Y., Yang, F., & Bi, W. (2018). A Spontaneous Facial Expression Database for Academic Emotion Inference in Online Learning. IET Computer Vision, 13. https://doi.org/10.1049/iet-cvi.2018.5281
6. W. Li, M. Li, Z. Su and Z. Zhu, "A deep-learning approach to facial expression recognition with candid images," *2015 14th IAPR International Conference on Machine Vision Applications (MVA)*, 2015, pp. 279-282, doi: 10.1109/MVA.2015.7153185.
7. A. Fathallah, L. Abdi and A. Douik, "Facial Expression Recognition via Deep Learning," *2017 IEEE/ACS 14th International Conference on Computer Systems and Applications (AICCSA)*, 2017, pp. 745-750, doi: 10.1109/AICCSA.2017.124.
8. E. Varadharajan, R. Dharani, S. Jeevitha, B. Kavinmathi and S. Hemalatha, "Automatic attendance management system using face detection," *2016 Online International Conference on Green Engineering and Technologies (IC-GET)*, 2016, pp. 1-3, doi: 10.1109/GET.2016.7916753.
9. R. Hartanto and M. N. Adji, "Face Recognition for Attendance System Detection," *2018 10th International Conference on Information Technology and Electrical Engineering (ICITEE)*, 2018, pp. 376-381, doi: 10.1109/ICITEED.2018.8534942.
10. Fuzail, M., Nouman, H.M.F., Mushtaq, M.O., Raza, B., Tayyab, A. and Talib, M.W., 2014. Face detection system for attendance of class' students. *International journal of multidisciplinary sciences and engineering*, *5*(4).
11. N. A. Shah, K. Meenakshi, A. Agarwal and S. Sivasubramanian, "Assessment of Student Attentiveness to E-Learning by Monitoring Behavioural Elements," *2021 International Conference on Computer Communication and Informatics (ICCCI)*, 2021, pp. 1-7, doi: 10.1109/ICCCI50826.2021.9402283.
12. S. Chowdhury, S. Nath, A. Dey and A. Das, "Development of an Automatic Class Attendance System using CNN-based Face Recognition," *2020 Emerging Technology in Computing, Communication and Electronics (ETCCE)*, 2020, pp. 1-5, doi: 10.1109/ETCCE51779.2020.9350904.
13. D. Mery, I. Mackenney and E. Villalobos, "Student Attendance System in Crowded Classrooms Using a Smartphone Camera," *2019 IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2019, pp. 857-866, doi: 10.1109/WACV.2019.00096.
14. Thomas, C. and Jayagopi, D.B., 2017, November. Predicting student engagement in classrooms using facial behavioral cues. In *Proceedings of the 1st ACM SIGCHI international workshop on multimodal interaction for education* (pp. 33-40).
15. Low-level networking interface, last accessed 25-7-2022. [Online]. Available: https://docs.python.org/3/library/socket.html

16. Low-level threading API, last accessed 25-7-2022. [Online]. Available: https://docs.python.org/3/library/_thread.html
17. Dimitriadou, Eleni & Lanitis, Andreas. (2022). Using Student Action Recognition to Enhance the Efficiency of Tele-education. 543-549. 10.5220/0010868200003124.

Appendix 1

Paper Presented at the 14th Cyprus Workshop on Signal Processing and Informatics, 2022

An Integrated Approach for Visualizing Student Activity During Distance Education

Michalis Kontos
CYENS Centre of Excellence
kontosmichalis24@gmail.com

Eleni Dimitriadou, Lefteris Ioannou
Visual Media Computing Lab,
Cyprus University of Technology
ela.dimitriadou@edu.cut.ac.cy, lyioannou@gmail.com

Andreas Lanitis
CYENS Centre of Excellence
& Cyprus University of Technology
andreas.lanitis@cut.ac.cy

***Abstract*—Distance learning became extremely popular during the COVID-19 pandemic, with many people working and attending classes from home. Due to privacy issues, in many countries the use of webcams is forbidden during tele-education. However, when the educator has no optical contact with his/her class, students tend to lose their concentration, and the overall teaching process is jeopardised. The aim of our work is to provide a system that will allow teachers to get informed about student activities during tele-education but without having access to video input from students. In the proposed system, images of students captured by webcams are processed locally at a students' personal machine, in order to identify the student actions. The relevant information is send to a server at the teacher's side. The server side contains a 3D visualization of the class where each student is represented by an avatar, animated to reflect student actions. This method will keep the teacher informed of the students actions without violating privacy barriers. Figure 1 shows a block diagram of the proposed system that includes the client (student side), and the server (teacher side). The work presented in this paper is influenced by previous work in student action recognition [1], [2], [3], and extends our previous work in the area [4].**

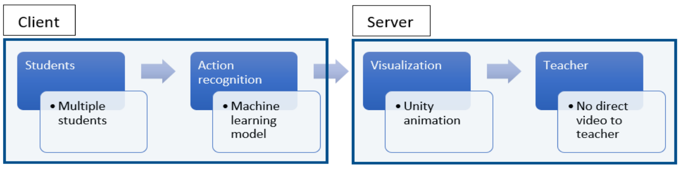

Fig. 1. System Overview

The client side of the proposed system (see Figure 1) incorporates a machine learning model trained to classify the following seven student actions: absent, attending, hand raising, looking elsewhere, telephone call, using phone and writing. The machine learning model used is based on the GoogleNet architecture [5], and during an experimental investigation it managed to classify correctly 94.32% image frames of previously unseen students [4]. The use of the GoogleNet model ensures that the computational requirements for the classification process are minimal, allowing the real-time operation on almost any personal machine used by students during tele-education.

Both the student and the teacher side use a python script for sending and receiving information about student actions. This is done using python socket [6] which provides a way for two nodes to communicate on a network. On the client side, classifications are performed constantly, sending information to the server side about the student action in each image frame captured. This is done at a rate of approximately 90 frames per 10 seconds. On the server side, the classifications are saved and on every iteration (10 seconds) a majority voting strategy within the selected time interval, is used to determine the exact student action to be written in a log file for each student.

At the server (teacher) side a Unity3D program reads the log file for each student, and activates the corresponding animation of each student avatar that matches the recognized student action. This process is done simultaneously for all students in class, so that the teacher visualizes the actions of all students taking place in a class in a similar way that he/she visualizes students in a real class. Figure 2 shows a screenshot of animated students in the virtual 3D class, where the actions of the student avatars correspond to the actions of each student at the client sides.

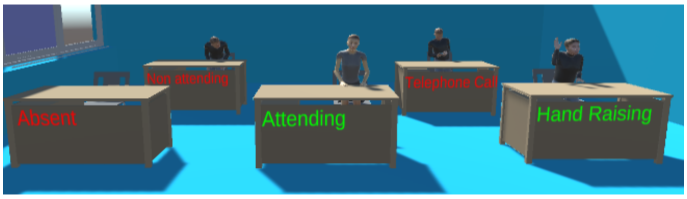

Fig. 2. Screenshot of animated students in virtual 3D class

Our work concerns the design of an integrated system where the teacher can visualize student actions, without affecting the privacy of students. Early results and feedback received by stakeholders, prove the promise of this approach. In the future, we aim to perform a comprehensive evaluation of the system, add in the recognition model the ability to recognize additional student actions and emotions, and also provide the ability for the teacher and students to view the class on an online multi-user virtual space, using appropriate Virtual Reality equipment.

References

[1] D. Mery, I. Mackenney, and E. Villalobos, "Student attendance system in crowded classrooms using a smartphone camera," in *2019 IEEE Winter Conference on Applications of Computer Vision*, 2019, pp. 857–866.

[2] C. Thomas and D. B. Jayagopi, "Predicting student engagement in classrooms using facial behavioral cues," in *Proceedings of the 1st ACM SIGCHI international workshop on multimodal interaction for education*, 2017, pp. 33–40.

[3] B. Snehitha, R. S. Sreeya, and V. M. Manikandan, "Human activity detection from still images using deep learning techniques," in *2021 International Conference on Control, Automation, Power and Signal Processing (CAPS)*, 2021, pp. 1–5.

[4] E. Dimitriadou and A. Lanitis, "Using student action recognition to enhance the efficiency of tele-education." in *VISIGRAPP (5: VISAPP)*, 2022, pp. 543–549.

[5] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 1–9.

[6] Low-level networking interface, last accessed 1-7-2022. [Online]. Available: https://docs.python.org/3/library/socket.html