A Generic AI-Based Technique for Assessing Student Performance in Conducting Online Virtual and Remote Controlled Laboratories

AHMED M. ABD EL-HALEEM1, MOHAB MOHAMMED EID2, MAHMOUD M. ELMESALAWY1, (Member, IEEE), AND HADEER A. HASSAN HOSNY3

1Electronics and Communications Engineering Department, Faculty of Engineering, Helwan University, Cairo 11795, Egypt
2School of Engineering and Applied Sciences, Nile University, Giza 3247010, Egypt
3Computer and Systems Engineering Department, Faculty of Engineering, Helwan University, Cairo 11795, Egypt

Corresponding author: Ahmed M. Abd El-Haleem (ahmed_abdelkhaliq@h-eng.helwan.edu.eg)

This work was supported in part by the Academy of Scientific Research and Technology (ASRT) through the Egypt Program (from 2021 to 2023) under Grant RDI/P4/1/20-21.

ABSTRACT Due to the COVID-19 pandemic and the development of educational technology, e-learning has become essential in the educational process. However, the adoption of e-learning in sectors such as engineering, science, and technology faces a particular challenge as it needs a special Laboratory Learning Management System (LLMS) capable of supporting online lab activities through virtual and controlled remote labs. One of the most challenging tasks in designing such LLMS is how to assess a student’s performance while an experiment is being conducted and how stuttering students can be automatically detected while experimenting and providing the appropriate assistance. For this, a generic technique based on Artificial Intelligence (AI) is proposed in this paper for assessing student performance while conducting online labs and implemented as a performance evaluation module in the LLMS. The performance evaluation module is designed to automatically detect the student performance during the experiment runtime and triggers the LLMS virtual assistant service to provide struggling students with the appropriate help when they need it. Also, the proposed performance assessment technique is used during the lab exam sessions to support the automatic grading process conducted by the LLMS Auto-Grading Module. The proposed performance evaluation technique has been developed based on analyzing the student’s mouse dynamics to work generally with any type of simulation or control software used by virtual or remote controlled laboratories; without the need for special interfacing. The study has been applied to a novel dataset built by the course instructors and students simulating a circuit on TinkerCad. Using mouse dynamics fetching, the system extracts features and evaluates them to determine if the student has built the experiment steps in the right way or not. A comparison study has been developed between different Machine Learning (ML) models and a number of performance metrics are calculated. The study confirmed that Artificial Neural Network (ANN) and Support Vector Machine (SVM) are the best models to be used for automatically evaluating student performance while conducting the online labs with a precision reaching up to 91%.

INDEX TERMS Artificial intelligence, mouse dynamics, laboratory learning management system, virtual assistant, artificial neural network (ANN), support vector machine (SVM).

I. INTRODUCTION

The ultimate goal of digital learning is to provide a platform for a broader range of people to achieve large learning outcomes [1]. In today’s world, the phrase e-learning, sometimes known as digital learning attracts many researchers. The distribution of learning content (e.g. text materials, audio, video, assessment, and interactive methods) through any electronic media is referred to as e-learning. In comparison to traditional face-to-face learning, this can aid to boost learning
levels. Various variables have been studied during the last decade to increase the performance of e-learning systems. These variables are the characteristics that can influence students’ happiness and enhances the prediction of digital learning systems [2].

E-learning does not allow students and professors to communicate directly. As a consequence, teachers face many difficulties with e-learning systems. From these difficulties is how the teacher will evaluate the course’s effectiveness on students. Moreover, from the difficulties that faces the teachers, is that the students in e-learning courses can dropout more than students in traditional learning modes. Furthermore, it is difficult to evaluate a student’s performance. Also, it’s difficult to forecast at-risk pupils in new classes. Finally, of the difficulties that face teachers is how to estimate their students’ projected outcomes on forthcoming tests [3], [4].

Due to the previous mentioned difficulties that faced the e-learning systems, many methods appeared to face these difficulties. One of the most effective recent methods is using Educational data mining (EDM) and Learning Analytics (LA) [5] to predict the student’s performance. EDM and LA are closely connected fields that attempt to aid in the analysis of educational data, due to the ever-growing data quantities which become increasingly difficult to process. As a consequence, EDM and LA offer tools and approaches for gathering, processing, and analyzing educational data. Most of the time, data mining, machine learning, and statistical analysis approaches use historical data to detect students’ performance.

As mentioned previously most of the EDM and LA depend on historical data. As a consequence, the researchers have focused on a variety of data elements, including past exam results, background, demographic data, and the students’ behaviours in and outside the classroom [6] to predict student success. The majority of performance prediction research focuses on prior exam results. However, relatively little research appears to focus on data analysis, such as logging and analyzing student interactions with online platforms [7].

It is found that using past scores on data analysis has two drawbacks: The first one is predicting the long-term performance of students from old collected data, which sometimes be inaccurate. For example, predicting a student’s performance in junior or sophomore year courses based on exam grades and course grades from previous years. The second one is if they take grades from major1 and major2 exams to predict the course performance, it is too late to perform any corrective actions to help the student succeed this semester. As a consequence, in this paper, a new generic technique is proposed to automatically evaluate lab asks based on the mouse dynamics of the students, not the historical data of students.

Furthermore, applying the performance evaluation module in the new LLMS provides an assistant, students friendly system, and simulates the atmosphere of a realistic laboratory. The system is tested with a customized dataset collected by the course instructors and teaching assistants, which is a record of mouse dynamics done by the student. Then, the feature extraction phase takes place to process data and provide meaningful information provided by the mouse actions. The extracted features will be introduced and processed by various Machine Learning (ML) Models to have a comparative analysis aiming to provide the best AI architecture for the task. Based on this, the contribution of this paper is summarized as follows:

- Design an effective LLMS that supports and manages various types of online labs such as virtual simulated labs and remote-controlled labs.
- A new intelligent and automatic Evaluation Module is proposed as the main part of the new LLMS, which helps in assisting the stumbled students while performing the experiments by detecting the stumbled parts of the experiment and providing them with the correct help. Furthermore, it provides the intelligence of evaluating the steps of the experiment while performing the experiments in the online practical exams.
- The proposed Performance Evaluation Module is designed to be generic, so that it can works with any simulation or control software for virtual and remote controlled lab experimentations.
- The newly designed generic Performance Evaluation Module is based on the idea of collecting the mouse dynamics and using the AI algorithms for evaluating student performance during experiment run-time.
- To the best of our knowledge, the proposed students’ performance evaluation technique, is a new generic technique that provides a general interface with any type of lab software and remote controlled experiments at the least cost.

The paper is organized as follows: Section I presents an overview introduction of the paper. Section II includes the previous literature review. Section III, explain the newly proposed LLMS and performance evaluation module in detail. Section IV presents the experiments carried out on a number of students to assist them while performing the experiment remotely. Section V presents a survey study that is performed on a number of students to evaluate the new system. Finally, section VI presents the conclusion.

II. LITERATURE REVIEW

Most of the approaches that have been developed for student performance prediction during performing an assignment were based on the student’s historical data. Many data are recorded for students such as the student’s previous grades and their overall performance are recorded. Although these methods helped to improve the student’s performance,
they couldn’t automate the process of evaluation of the assignment.

Buenaño-Fernández, et al. [8], proposed a new research paper that predicts the student’s performance based on previous and current exam grades. The target is to identify the students’ “at-risk” for each subject and to provide real-time feedback on their current status. Such information would help in the development of appropriate remedial strategies to help these students and improve their retention rates. They used the Decision Tree algorithm in the prediction model of their work to classify students into passing and failing categories.

S. Huang and N. Fang [9] concentrated on forecasting student success in engineering dynamics. The authors used four prediction models in their study: multiple linear regression (MLR), multilayer perceptron network (MLPN), radial basis function network (RBFN), and support vector machine-based model (SVM). The goal of their research was to find the best prediction model and the most influential variable within a group of six variables that could lead to a more accurate prediction model. Only historic performance data were included in this study’s dataset. It covered nine different grade dynamics as well as pre-requisite dynamic courses. This model, like the one described in [8] only looked at one type of student data (past performance-related data) and didn’t take into account other factors like engagement or demographics.

On the other hand, some researchers such as Huan Wei, et al. [10] proposed new features based on student mouse movement trajectories to define students’ problem-solving characteristics (e.g., think time, first try, and first drag-and-drop). Furthermore, a heterogeneous information network is used to integrate students’ previous problem-solving knowledge on comparable topics in order to improve the student performance predictions on a new question. The authors used four Machine learning models. The results demonstrate that approached methodology can predict student performance in interactive online question pools with higher accuracy than the usual method of merely employing statistical variables (e.g., students’ history question scores) in various models.

S. Liu and M. d’Aquin [11], used unsupervised learning to see how students’ demographic factors and participation in online learning activities impact their learning outcomes. Their approach used the K-Prototypes clustering approach to identify student groups based on demographic features and interactions with an online learning environment. They used the results gathered for each group of students regarding their positive or negative learning outcomes to create online courses that cater to the diverse student’s requirements. Furthermore, it helps the students to choose online courses that are most compatible for them.

One of the interesting approaches has been adopted by Brahim [12], which used a real dataset acquired using a Digital Electronics Education and Design Suite (DEEDS) [5] to predict students’ performance throughout a series of online interactive sessions. The dataset records student engagement during online lab work in terms of text editing, keystrokes, time spent on each activity, and the test score earned every session. The suggested prediction model is made up of 86 new statistical variables that were semantically classified into three major groups based on criteria including activity type, timing statistics, and peripheral activity count. During the feature selection step, this collection of characteristics was further reduced, and only the most influential elements were kept for training. The suggested machine learning approach is designed to predict whether a student’s performance would be poor or excellent. Random Forest (RF), Support Vector Machine (SVM), Nave Bayes, logistic regression, and multilayer perceptron were the five common classifiers employed in their research.

As presented in the previous research their work is based on predicting student performance, most of the contribution has been made based on student personality, historic performance, and platform interactions to determine the certainty of the student about his answer. Unfortunately, this may not help in determining the validity of the task outcome itself as most of the data and features extracted are based on the student only. As a consequence, to perform the task evaluation automatically and validation of students’ answers, the research started to concentrate on another form of data which is mouse dynamics during performing the task. Many researchers have contributed to the use of mouse dynamics as data for AI models but not to evaluate tasks. Mostly, they were discussing the user authentication based on his mouse dynamics during sessions.

From the recent research that used the mouse, dynamics is the one proposed by T. Hu, et. al [13]. He used mouse dynamics deep learning as an approach for performing continuous user identification effectively and quickly while addressing insider threats. The research was based on an open-sourced dataset which is the Balabit Mouse Dynamics dataset [14]. Experiments were carried out with 10 users. The dataset was mapped in the shape of an image to represent users’ behavior, then they were introduced to a CNN model to be trained with. The findings confirmed the usefulness of the strategy. Achieved a false acceptance rate of 2.94 percent, a false rejection rate of 2.28 percent, and an authentication time of 7.072 seconds. These findings indicate that this method was capable of detecting insider threats in a particular setting.

M. Antal and N. Fejér [15], presented a new convolutional neural network for extracting characteristics from user mouse movement time series. They used several training modes, such as transfer learning and training from scratch were used in their model. Both authentication and identification system results are evaluated using the performance metric Receiver Operating Characteristics (ROC). The ROC curve is a graph that shows how well something works. The area under the curve (ROCAUC) is frequently used to compare the performances of different biometric systems, and it compares the true positive ratio (TPR) against the false positive ratio (FPR). The Balabit public dataset [13] was utilized for performance evaluation, however, the DFL data set [16] was used for transfer learning. From the gathered results, it is found that the novel used 1D-CNN model outperforms the other CNN
models presented for the same job, especially with using transfer learning.

It has been noticed also that the sequence of events and actions in time series affects the prediction process. Accordingly, the authors in [17] have implemented a new approach in user authentication using LSTM and CNN-LSTM hybrid model aiming to take advantage of the sequential form in data related to mouse dynamics. The research has experimented with those models using the Balabit Mouse Dynamics Challenge [14] and the Wolf of SUTD (TWOS) [18] datasets. Convolutional networks, recurrent networks, and a hybrid model combining convolutional and recurrent layers are used in the study to classify mouse movement sequences. It is well known that training these networks on small datasets with random initialization of weights results in models that perform badly.

As a consequence, the authors explore a two-dimensional convolutional neural network using transfer learning, which is a domain adaptation strategy that works well with tiny datasets. Although using such a design may appear contradictory, it has outperformed all other deep architectures tested, as well as a traditional machine learning approach, because the temporal information is eliminated from the input data. To compute relevance scores for each component of the mouse curves, the suggested approach used the layer-wise relevance propagation (LRP) technique. The models are also evaluated in terms of their usefulness and efficacy in real-world circumstances.

III. PROPOSED SYSTEM

The proposed system describes first the new intelligent generic Laboratory Learning Management System (LLMS) with a performance evaluation module and virtual tutor. Afterward, the operation of the virtual assistant and Lab evaluation component that exists in LLMS is described in detail.

A. THE PROPOSED GENERIC LLMS

The new intelligent generic LLMS provides many features that help the students and teachers while assigning and performing experiments online. One of the main features of the new LLMS system is the performance evaluation module, which automatically triggers the virtual assistant service to detect if the students need help or not and show suitable help while performing online experiments. This new feature is provided in the new LLMS system with the intelligence of providing help to students in a suitable form such as text, audio, or video help according to the student’s cognitive style [19]. Furthermore, if the students need extra help by performing chatting with the tutor or colleague, the session collaborator’s help from audio, text, and video are used [19]. Also, the performance evaluation module helps in evaluating the conducted experimental steps of students while performing the practical exams by collecting the mouse dynamics behavior of the students and triggers the auto-grading module to evaluate the steps of the experiment performed by the students.

The architecture of the new generic LLMS system is presented in Fig. 1. The LLMS system is proposed to allow the teacher and students to assign and perform virtual and remote controlled lab experiments and practical exams online.

Furthermore, the new LLMS system provides teachers with the simplicity of designing the hardware and software experiments remotely. The new LLMS system added the performance evaluation module that triggers the virtual assistant service, which acts as a virtual tutor that helps the students while performing the experiments. As the performance evaluation module detects that the student needs help while performing the experiment, and then triggers the virtual...
The performance evaluation module is used to provide the intelligence of the new LLMS system by learning from the behavior of students performing the parts of the experiment if they need help triggers the e-learning virtual assistant service to provide the students with suitable help or open chatting with other using the Session Collaborator while performing an experiment using the mouse dynamics behavior.

The Experimental View is used to preview the shape of the designed experiment or practical exam interface while designing them using the authoring tool.

After designing the experiment or the practical exam using the authoring tool, the generated LLO objects stored in the LLO store of the LLMS database are scheduled and assigned to one of the available resources in the laboratory server using the Lab Resource Manager by the Experiment Runtime Engine Module.

The students access the assigned experiment or practical exam through the internet remotely by using the Students Remote Access Interface, which is used to read the LLO assigned to the student from the LLMS database using the Experiment Runtime Engine module. Furthermore, the Experiment Runtime Engine module will run the Resources Manager to assign the user one of the VM or physical machines that exists on the server. If the LLO includes the e-learning virtual Assistant component, the Experiment Runtime Engine module will run the performance evaluation module which collects the mouse dynamics and store it in the LRS store, then triggers the virtual assistant service to act as a virtual tutor. However, if the students are performing a practical exam, the performance evaluation module which collects the mouse dynamics will evaluate the experimental steps performed by students and triggers the auto-grading module to perform auto-grading for students’ exams.

Finally, after the students performed the experiment and all the required added services are run, the output of the experiment will be stored in an LRO (Learning Report Outcome) and stored in the LRO store of the LLMS database student’s profile. If the students finished performing a practical exam, the answers and the student’s evaluation of performing the experimental steps will enter an auto-grading module, to mark the students’ answers and store both the answers and the student’s performance evaluation system works. The main function of this experiment is to have a house intruder system along with a motion detector for the home light system.

The new LLMS’s performance evaluation Module provides the system with the new intelligent virtual assistant service and auto-grading components. The main idea of the performance evaluation Module is to collect the students' mouse dynamics while performing the experiment in the runtime using the Experiment Runtime Engine Module. Afterward, those mouse dynamics are stored in the Learning Record Store (LRS) in the LLMS database. The LRS stores all the learning information associated with students as defined by the xAPI spec documentation. Afterward, the performance evaluation Module runs the AI algorithm to detect if students need help. Then it triggers the virtual Assistant service to read the stored mouse dynamics from LRS through Experiment Runtime Engine Module to automatically show the correct help for the students while performing the experiment. Furthermore, the performance evaluation Module runs the AI algorithm to detect if students performed the experimental steps correctly while performing a practical exam and triggers the auto-grading module to automatically evaluate and grade the experimental steps performed by the students.

As shown in Fig. 1, the new generic LLMS system actors are the teacher and students. The teacher accesses the LLMS’s system through an internet connection. Afterward, the teacher starts to design the online experiment or exam using the authoring tool. The new authoring tool consists of the following:

- The Generate Experiment Content Package is used to design the experiments or exams.
- The Content Component Assets are used to add static draggable components for experiments such as text, images, tables, etc.
- The VRL-specified Component Assets are used to add the simulator component as part of experimental exams or experiments by automatically and remotely accessing the VM using the Myrtille technology. Moreover, the VRL-specified Component Assets are used to open sessions to control the hardware that exists physically in the laboratory.

The Lab Learning Object (LLO) represents the generated designed experiment or practical exam by a teacher, stored as a JSON object, which is stored in the LLMS database.

The new LLMS’s performance evaluation Module provides the system with the new intelligent virtual assistant service and auto-grading components. The main idea of the performance evaluation Module is to collect the students’ mouse dynamics while performing the experiment in the runtime using the Experiment Runtime Engine Module. Afterward, those mouse dynamics are stored in the Learning Record Store (LRS) in the LLMS database. The LRS stores all the learning information associated with students as defined by the xAPI spec documentation. Afterward, the performance evaluation Module runs the AI algorithm to detect if students need help. Then it triggers the virtual Assistant service to read the stored mouse dynamics from LRS through Experiment Runtime Engine Module to automatically show the correct help for the students while performing the experiment. Furthermore, the performance evaluation Module runs the AI algorithm to detect if students performed the experimental steps correctly while performing a practical exam and triggers the auto-grading module to automatically evaluate and grade the experimental steps performed by the students.

The total lab is divided into a number of sequenced tasks. After each task, if the student did not solve it right, the system will provide a hint for him to help. So, there are a specific
number of checkpoints for the lab, after each checkpoint, the system evaluates student steps and according to this, it provides a hint. In case he solved the lab incorrectly again, the system provides another hint and this process will be looped till there is no hint left. Then the system will advise the student to have a meeting with the teaching assistant associated with the course.

Based on Fig. 2, the dataset for mouse dynamics for a number of students performing an online lab task on LLMS is collected, then the process of action segmentation is performed to divide the collected mouse dynamics of the students while performing the lab experiments into a number of actions. Next, the collected mouse actions will enter the extraction stage to extract the main features as will be explained later. Afterward, the lab evaluation system is used to detect if the students performed lab experiment steps (i.e., tasks) correctly or not. Then, a number of AI algorithms are used in the prediction, to detect if students performed steps correctly or need help. Finally, if the students perform an exam so an evaluation is performed by the performance evaluation system, and the grade is approved, and add results through the auto-grading Module to LRO as explained in the previous section. However, if Manual Grade Validation is needed so use the teacher assistant or instructor to perform grading and store the result in the LRO. The reason behind choosing the mouse dynamics to evaluate the practical lab was due to the hardware dependency in using unstructured data such as images. According to Xin, et al. [20], one of the main advantages of using traditional machine learning techniques is the low computational power needed for it in relation to complex deep learning models such as CNNs, LSTMs, Hybrid models and reinforcement learning. Therefore, this gives traditional machine learning an edge over Complex models with respect to hardware dependency. That is why the whole research and the proposed system is based on mouse dynamics instead of images or unstructured data, to reduce the hardware dependency. In the following subsection, each process can be discussed explicitly and in more detail.

1) B.1 DATASET
A lab experimental experiment built on the new LLMS as presented in Fig. 3 is divided into a number of pages (i.e., tasks), each task consists of a number of steps and acts as a separate experiment that depends on the previous task and with increasing hardness. Since there were no available datasets that simulate a general experimental task without depending on specific software. As a consequence, it was a necessity to build such a dataset [21] to support the generic feature of the new LLMS. The lab experimental task that was suggested in this research was based on TinkerCad online circuit simulator [22]. The TinkerCad simulator was used in the experiment to train the student to build a low-level circuit design and to evaluate the student’s concepts regarding building the circuit. The purposed task scenario in Fig. 3 for the lab experimental task is as follows:

- Grab the Arduino Uno to the simulation area and attach a LED to it, then connect the Anode pin to Pin 13, and the Cathode side to GND.
- Grab the PIR sensor to the simulation area and attach it to the Arduino, then connect the signal pin to Pin 8, the power pin to 5V, and the ground pin to GND.
- Grab the ultrasonic sensor and attach it to Arduino, then connect the GND pin to GND, VCC pin to 5V, and SIG pin to Pin 6. Afterward, grab the Buzzer and attach it also to the Arduino by connecting the Positive pin to Pin 7, and the Negative pin to GND.

Each page (i.e., stage) of the lab experiment has been evaluated separately as a task using the LLMS. The inputs that have been provided as data to the models are the mouse dynamics which are the Clicks, Moves, and Drags. When the student is assigned an experiment, so the Experiment Runtime Engine Module in the new LLMS will run a new python script [23] on the student’s machine to fetch activities made by the mouse and gives all details about it such as the time when the action was taken which are represented by record Time, and client Time. Also from the fetched activities, details are the coordination of x and y of the mouse during this action. It is important to mention that the X-coordinates and Y-Coordinates provided by the system are adjusted with the standard ratio for the common PC screens 16:9 to fit in the resolution of 1920:1080, to avoid the difference in screens resolutions among all PCs. Furthermore, the button used for the action (i.e., button), and the type of action itself (i.e., state)
are from the fetched data. A sample of these collected raw data from the python script is presented in Fig. 4.

The dataset was divided into 3 partitions, which are task 1, task 2, and task 3, each with more than 600 instances of actions made. All three tasks are stages to construct the practical lab as a whole project.

After collecting the Raw data as presented in Fig. 2, the data need to be reconstructed to a more meaningful form for further processing using Action Segmentation. This was accomplished by converting all the collected raw data instances to action events. For example, the movement of the mouse between two screen coordinates is described by a mouse action, which is a sequence of sequential mouse events. Because each event includes a screen position, a mouse action with n events is represented as a series of n points. Each point is representing an instance. Each session’s data was divided into three categories of mouse actions. Ahmed and Traore [24] identified these action kinds as MM, PC, and DD. A general mouse movement action is denoted by the letter MM. The mouse pointer is simply moved between two spots on the screen in this operation. PC stands for point and click. This motion is similar to mouse movement, except it concludes with a mouse click. Drag and drop action is denoted by the letter DD, and it always begins with a left mouse button pushed event, then a series of mouse drag events, and finally a left mouse button released event.

For example, in a PC-type action, the segment’s ending is marked when the last event of a mouse-released event is a mouse-pressed event. Otherwise, a DD-type action is used to signal the end of a segment. Before concluding in a PC or DD-type action, these segments may include one or more MM-type actions. The MM activities are segregated in the second phase using a time field threshold. When the time gap between two consecutive events exceeds a threshold, the segment is split into two actions. The time limit has been set at 10 seconds. Finally, actions with less than four occurrences were disregarded using the spline interpolation, which requires at least four points.

Finally, after collecting Raw data and performing Action segmentation now features can be extracted as will be presented in the next subsection.

2) B.2. FEATURE EXTRACTION

The feature extraction is performed to extract the main features needed to train the AI algorithms. As mentioned in the previous subsection Action segmentation is used to extract the mouse actions from raw data. Afterward, the feature extraction is used to extract statistical features for each action segment, the research has adopted the approach that has been studied by Gamboa and Fred in [25], as they used the 3 data rows only from each action segment which are Client time, x and y coordinates. However, buttons and states were used in the segmentation process. Hence each action is represented as a sequence of \((t_i, x_i, y_i)\) for each \(i\) in \(n\), where \(n\) is the number of mouse records in each segment.

The angle of the path tangent with the x-axis was calculated using the sequence of \((x, y)\) by (1):

\[
\alpha_i = \text{atan}2 \left( \frac{\delta y_i}{\delta x_i} \right) = \arctan \left( \frac{\delta y_i / \delta x_i}{\delta x_i / \delta x_i} \right)
\]

where \(y_i = y_i - y_{i-1}, x_i = x_i - x_{i-1}, \alpha_1 = 0\), and the atan2 trigonometric function is a common variation of the standard arctangent function. Then the features: \(v_x, v_y, v, a, j, \) and \(w\), which are horizontal velocity, vertical velocity, velocity, jerk, and angular velocity, respectively were calculated based on the statistical feature extraction method proposed by [24] using (2) and (3):

\[
v_x = \frac{\delta x_i}{\delta t_i}, v_y = \frac{\delta y_i}{\delta t_i}, v = \sqrt{v_x^2 + v_y^2}
\]

\[
a_i = \frac{\delta v_i}{\delta t_i}, j_i = \frac{\delta a_i}{\delta t_i}, w_i = \frac{\delta a_i}{\delta t_i}
\]

where \(\delta a_i = a_i - a_{i-1}, \delta v_i = v_i - v_{i-1}, \delta a_i = a_i - a_{i-1}, a_1 = j_1 = w_1 = 0\).

One of the features that have been calculated is the curvature time series, which is the ratio between the angle change to reduce the direction of each action segment, the method of

\[
s_i = \sum_{k=1}^{i} \sqrt{x_k^2 + y_k^2}
\]

\[
C_i = \frac{\delta s_i}{\delta t_i}
\]

where \(s_i = s_i - s_{i-1}, s_1 = 0\), while the critical point of \(C\) is the low Curvature point with a threshold \(TH_c = 0.0005\). Each statistical feature has been converted to four features, which are Mean Standard deviation, Minimal value, and Maximal value to be used as the main features of the action segment.

Besides those features, there were other features which are the type of action segment, and the length of the trajectory \(s_n\) length of the segment between two endpoints. Also, the time needed to perform this action is considered a feature. Aiming to reduce the direction of each action segment, the method of
eight main directions, which was defined by [24], has been used.

There were other features which are the number of points, which represents the total number of mouse events in an action segment, the sum of projections, and the maximum deviation, which is the segment between the two endpoints, and the distance separating the trajectory’s points that is the greatest. Also, the length of the trajectory divided by the segment between an action’s two endpoints was used to calculate the straightness. Finally, the duration of the first segment of the action with positive acceleration was calculated to be used as a feature. A summary of the 39 features extracted is presented in Table 1.

| Feature | Description          | No. of parameters |
|---------|----------------------|-------------------|
| $v_x$   | Mean, Std, Max, Min  | 4                 |
| $v_y$   | Mean, Std, Max, Min  | 4                 |
| $v$     | Mean, Std, Max, Min  | 4                 |
| a       | Mean, Std, Max, Min  | 4                 |
| j       | Mean, Std, Max, Min  | 4                 |
| w       | Mean, Std, Max, Min  | 4                 |
| c       | Mean, Std, Max, Min  | 4                 |
| Type    | MM, PC, DD           | 1                 |
| Elapsed time | $t_e - t_i$   | 1                 |
| Trajectory length | $s_n$   | 1                 |
| End-to-End distance $| |$ | 1 |
| Direction | $[1,8]$   | 1                 |
| Straightness | $d$   | 1                 |
| No. of points | $n$   | 1                 |
| Sum of angles | $\sum_{i=1}^{n} g_i$ | 1 |
| Largest deviation | $\max(d(p_i, [p_1 \cdot p_n]))$   | 1 |
| No. of Critical points $| |$ | 1 |
| a Beginning time | Acceleration at beginning | 1 |

| Total features | 39 |

3) B.3. DATA PREPROCESSING

After collecting the features, data preprocessing is performed to make sure that there is no mislabeling and to check the datatype of each feature. The number of actions that will be processed has been reviewed to consider the misbalancing of the data. The number of corrective actions to accomplish the lab task was 993 samples, while the number of the overall actions with wrong actions was 1027.

Moving from checking the imbalanced data, the distribution of feature values was concerned to clear any outliers in the data. Outliers are checked by making sure that there is a normal distribution for feature values. This can be checked using fast analysis as knowing the mean of the feature values, Min., Max., standard deviation, and percentiles. After that, the correlation between features was introduced to make sure that there is no high correlation between features. High correlation may cause overfitting in the model as it causes neural representations to become more redundant and can therefore reduce the expressive ability of neural networks. One may explicitly penalize the correlation between features as part of the objective function to eliminate this impact but doing so comes at a significant computing cost [26]. The correlation between the weight vectors (or convolutional kernels) of various hidden units may also be penalized [27], but this method is less computationally expensive and only slightly more effective.

Furthermore, both strategies need the penalty strength to be manually adjusted. Instead of combating feature association with additional regularization methods, it would be preferable to just avoid it in the first place by dropping the highly correlated features. The correlation between two features is calculated using (6).

$$r = \frac{\sum (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum (x_i - \bar{x})^2 \sum (y_i - \bar{y})^2}}$$

where $r$ is the correlation coefficient, $x_i$, $y_i$ are the values of features $x$ and $y$ in $i^{th}$ sample, and $\bar{x}$, $\bar{y}$ are the mean value of features $x$ and $y$ for the overall data.

The correlation table was visualized in Fig. 5. As it appears, there is a high correlation between the standard deviation of $v$ and $v_x$, so the feature that represents the standard deviation of $v_x$ has been dropped. Also, there was a high correlation between the standard deviation of $v$ and the maximal value of $v$, so the maximal value for $v$ was dropped. One of the most noticed high correlations was between features extracted from acceleration and jerk $a$, $j$ respectively. As there was a high correlation between the mean, standard deviation, and maximal value of $a$, with standard deviation and maximal value for $j$. So mean of $a$ was chosen to be in the features that are being used in the prediction and all other features were dropped, and it can be noticed the final form of features will be used in the prediction model as an input can be presented in Fig. 6.

After choosing the feature for the model, the data had to be divided between the training set and the testing set. As the whole samples are less than 2500, the data was divided with 70:30 ratios between training and testing, respectively. Finally, the whole data has been normalized using the standard scaler method as in (7).

$$z = \frac{x - \mu}{\sigma}$$

where $x$ is the current value, $\mu$ is the mean value of all samples, and $\sigma$ is the standard deviation of all samples.

The reason behind using the standard scaler is to make it easier for the model to predict the pattern between data. The difficulty of the task being modeled might be made more difficult by differences in scales across input variables. As an illustration, a model may learn huge weight values when given big input values such as a range of hundreds.
or thousands of units. A model with high weight values is frequently unstable, which means that it may exhibit poor learning performance and sensitivity to input values, leading to a larger generalization error. So, normalizing data using a standard scaler reduces the scale of the data which leads to the stability of the model.

4) B.4. PREDICTION MODELS
In this section, the different prediction models used to train the collected raw data after the previous steps are explained in detail. Each model will be specified with its hyperparameters and architecture, also they will be illustrated in detail.

5) SUPPORT VECTOR MACHINE
Support Vector Machines (SVM) is a class of statistical machine learning techniques, that aims to lower the structural risk principle. SVM’s feature gives it a strong foundation for use in a variety of pattern analysis issues [28]. Vapnik et al [29] introduced this class for a two-class categorization issue. The SVM was further expanded to address challenges involving several classes of categorization. The foundation of SVMC relies on preprocessing training data points and attempts to create a hyperplane that can accurately segregate training data points. To locate a new data point’s associated cluster, SVMC measures the distance between the data point and a chosen hyperplane. The labeled entries in the training data points denote the entry cluster. The kernel function is a distance function carried out in the measuring procedure. Several distinct Kernel functions are depending on the type of item and how it is distributed, whereas the Kernel function would measure the distance between objects. Like other kinds of learning techniques, the performance of SVMC is unquestionably influenced by the Kernel function and the training data points, which must accurately reflect the clusters. There are two different kinds of data points in any problem: those that can be separated nonlinearly and those that can be separated linearly. It is a known fact that non-linearly separable data points can be separated using a linear function by increasing the dimension of the data points. This fact is utilized by SVMC, which then applies a map function to inject data points from the Hulbert space into a higher-dimensional subspace, where a kernel function is intended to linearly divide the data points. To generalize the model, a soft margin was used. A soft margin is used to allow some misclassification in the data, hoping to have a more generalized model, and the value of this soft margin has been selected to be 2. The radial basis function (RBF) was used as a kernel to increase the dimensionality of the data, which will help the model to find the hyperplane.

6) NAIVE BAYES
One of the most well-known Bayesian classification algorithms used in Naive Bayes is the NBC (Naive Bayes Classifier) feature, which is a straightforward algorithm structure with a high level of processing efficiency. The Naive Bayes classifier has the benefit of just requiring a modest quantity of training data to estimate the essential parameters which are the variable mean and variance. The entire covariance matrix is not required because independent variables are assumed, just the technique of estimating each variable is needed. Using the maximum likelihood estimation principle, naive Bayes classification will classify the data into the category with the highest probability [29] as appears in (8).

\[
P(C_i|X) = \max \{P(C_1|X), P(C_1|X), \ldots P(C_n|X)\} \quad (8)
\]
While sample \( X = \{A_1, A_2, \ldots, A_K\} \) is an attribute \( A_j \), Naive Bayes classification considers that the attributes are independent of each other, so to calculate conditional probability, (9) will be used.

\[
P(X | C_i) = \prod_{j=0}^{k} P(A_j = x_j | C_i) \tag{9}
\]

Since the conditional probability of having a specific class given feature can be retrieved using (10), so (9) can be substituted in (10).

\[
P(C_i | X) = \frac{P(X | C_i) P(C_i)}{\sum_{i=1}^{n} P(X | C_i) P(C_i)} \tag{10}
\]

So, now the new formula for calculating \( P(C_i | X) \) can be represented in (11).

\[
P(C_i | X) = \prod_{j=0}^{k} \frac{P(A_j = x_j | C_i) P(C_i)}{P(X)} \tag{11}
\]

Then, let’s have a notation for \( \frac{1}{P(X)} = \alpha (> 0) \), so the final form for \( P(C_i | X) \) can be represented in (12).

\[
P(C_i | X) = \alpha \prod_{j=0}^{k} P(A_j = x_j | C_i) P(C_i) \tag{12}
\]

Now, (13) is the representation of the probability of having class \( C_i \), knowing that dataset \( D \) with a number of samples \( N(D) \), and the number of samples categorized to class \( C_i \) is \( N(C_i) \). While (14) is the representation of the probability of having an attribute in a given class, as \( N(C = C_i, A_j = x_j) \) represents the number of samples when attribute \( A_j \) is \( x_j \) in \( C_i \).

\[
P(C_i) = \frac{N(C_i)}{N(D)} \tag{13}
\]

\[
P(A_j = x_j | C = C_i) = \frac{N(C = C_i, A_j = x_j)}{N(C_i)} \tag{14}
\]

Finally, by substituting (13), (14) in (12), the final formula for calculating NBC can be presented in (15).

\[
P(C_i | X) = \alpha \prod_{j=0}^{k} \frac{N(C = C_i, A_j = x_j)}{N(C_i)} \cdot \frac{N(C_i)}{N(D)} \tag{15}
\]

Using this equation, NBC can be used in the classification problem of the study.

7) K-NEAREST NEIGHBOURS

In the case-based learning approach known as KNN (K-Nearest Neighbours), all training data is retained for classification as in [30], and by knowing the nearest sample classification, the prediction of the test sample can be determined. It cannot be used in many applications, such as dynamic web mining for a big repository, because it is a slow learning approach. Finding some representatives to represent the entire training dataset for classification is one strategy used to increase its effectiveness. Another method is to develop an inductive learning model from the training dataset and use this model (i.e., representatives) for classification. Numerous current techniques, like decision trees and neural networks, were first created to construct such a model. The performance of different algorithms is one of the evaluating criteria [31].

The fact that KNN is a straightforward but effective method for classifying data and that it is persuading as one of the best methods on the corpus of newswire stories from Reuters that motivates us. The main concept behind KNN is to measure the Euclidian distance between samples to classify based on the majority votes of samples clustered to a specific class using (16).

\[
d(p, q) = \sqrt{\sum_{i=1}^{n} (q_i - p_i)^2} \tag{16}
\]

While \( q_i \) is the \( i \)th feature in, \( p_i \) is the \( i \)th feature in \( p \). So, by comparing the test sample with all data provided to the model, the first \( K \) samples class majority determines the class of the test sample. A new implementation for KNN is presented in Alg. 1, in order to optimize computational power and be compatible with the proposed case study with a large dataset.

As it appears in Alg. 1, KNN consists of two main parts. First, calculates the Euclidian distance between the test sample and all data samples and sort them based on this distance. Then, collecting the votes refer to the classification of the test sample. Since the features extracted were of high dimensionality and complexity, it was needed to implement KNN from scratch with a simple modification to optimize the computational power needed for the model in large datasets. The \( K \) value that has been chosen for KNN was 8. This value was evaluated using 3-Folds cross-validation. The choice of 3-Folds was based on the splitting of data in the main division of the data which was 70:30 for training and testing.

**Algorithm 1 K-Nearest Neighbours**

**Input:** Dataset: \( D = \{X_1, X_2, \ldots, X_N\}, X_i = [x_{i1}, x_{i2}, \ldots, x_{id}] \), is the matrix of features representing sample \( i \), and the test matrix represented as \( X_{test} = [x_{t1}, x_{t2}, \ldots, x_{tk}] \), beside the \( k \) value that presents the number of neighbors that will be relay on to classify.

**Output:** The classification of the test sample.

1. for \((i = 1 \text{ to } n) \) do \( l/n \) is the number of data samples
2. for \((j = 1 \text{ to } k) \) do \( //l/k \) is the number of features
3. calculate \((x_{ij} - x_{jt})^2 // \text{equation 16} \)
4. add Value to \( sq \)
5. end
6. append \( \sqrt{sq} \) in \( \text{distance} \) array with class of \( X_i \)
7. end
8. sort \( \text{distance} \) array based on \( \sqrt{sq} \) value
9. for \((c = 1 \text{ to } K) \) do
10. increment element \( \text{class}_{\text{distance}[c]} \) in \( \text{class} \) array
11. end
12. return \( \text{argmax(class)} // \text{output index of the element with maximum value} \)

\( // \) end KNN algorithm. */
respectively. Cross-validation is a statistical technique for calculating the effectiveness (or accuracy) of machine learning models. When the amount of data is likely to be restricted, it is employed to prevent overfitting in prediction models. In cross-validation, the data are folded (or partitioned) into a predetermined number of folds, the analysis is performed on each fold, and the total error estimate is then averaged as in [32].

\* is the index of samples.

\*j is the index of features in each sample.

\*sq variable to save total squaring of difference between features of the test sample and data samples

\*distance is the array that stores Euclidian distance along with the class (dimension [2][n])

\*class is the array that stores the number of samples in each class in the nearest K samples.

Cross-validation is used commonly to tune hyperparameters of a machine learning model to achieve the best values that avoid overfitting without degrading model performance.

8) DECISION TREE

The decision tree is a dataset structure made up of leaves and decision nodes. One of the listed qualities that are picked at the node is indicated by a decision node. A leaf represents a class value. An internal node is displayed for each test result that might be obtained [33]. The Decision tree in this study was built based on the Gini impurity method by suggesting the optimal from root to nodes reaching to the leaves. Although the features have high dimensionality, the maximum depth for the decision tree is limited to 8 nodes for maximum depth. This value was chosen empirically to avoid overfitting in the model, as decision trees are normally tending to overfit. Providing a tree with high depth will lead to an unreliable model for real-life applications. After fitting the model to the data, the tree produced can be presented in Fig. 7.

9) RANDOM FOREST

Random forest (RF) is a decision tree mashup using majority voting to create a classification or regression prediction, which is an ensemble training approach. Due to their ease of use, adaptability, and interpretability, they are highly well-liked. RFs are fundamentally a vast array of if-else expressions. In an RF, the inputted value is “tested” at each node of a decision tree. The total sample space for the entered value is indicated by each branch at that node. Data is finely divided and further classified by the multiple decision trees as the entered value moves down the RF [34]. An RF’s
In most learning models, noise, bias, and variation are the major sources of mistakes. Therefore, there is a good possibility that any training technique that can effectively reduce these mistakes will increase the precision and robustness of machine learning algorithms. In order to reduce the mistakes from each model, ensemble training combines several decisions from many models. The algorithms previously mentioned are structurally independent and only use a single model’s classification decision to categorize a data point. The aggregate judgment of these weak classifiers frequently results in a moderate, averaged output, thanks to ensemble training, which allows several weak classifiers to be combined into a family. It is reasonable to assume that each family of poor classifiers will make mistakes while using ensemble training. It is done in such a way that the correlation between each set of weak classifiers is kept to a minimum.

As a result, the likelihood of a certain mistake may not be as common in other groups as it is in one group. Therefore, there is a good possibility that any training technique that can effectively reduce these mistakes will increase the precision and robustness of machine learning algorithms. In order to reduce the mistakes from each model, ensemble training combines several decisions from many models. The algorithms previously mentioned are structurally independent and only use a single model’s classification decision to categorize a data point. The aggregate judgment of these weak classifiers frequently results in a moderate, averaged output, thanks to ensemble training, which allows several weak classifiers to be combined into a family. It is reasonable to assume that each family of poor classifiers will make mistakes while using ensemble training. It is done in such a way that the correlation between each set of weak classifiers is kept to a minimum.

As a result, the likelihood of a certain mistake may not be as common in other groups as it is in one group. Therefore, there is a good possibility that any training technique that can effectively reduce these mistakes will increase the precision and robustness of machine learning algorithms. In order to reduce the mistakes from each model, ensemble training combines several decisions from many models. The algorithms previously mentioned are structurally independent and only use a single model’s classification decision to categorize a data point. The aggregate judgment of these weak classifiers frequently results in a moderate, averaged output, thanks to ensemble training, which allows several weak classifiers to be combined into a family. It is reasonable to assume that each family of poor classifiers will make mistakes while using ensemble training. It is done in such a way that the correlation between each set of weak classifiers is kept to a minimum.

As a result, the likelihood of a certain mistake may not be as common in other groups as it is in one group. Therefore, there is a good possibility that any training technique that can effectively reduce these mistakes will increase the precision and robustness of machine learning algorithms. In order to reduce the mistakes from each model, ensemble training combines several decisions from many models. The algorithms previously mentioned are structurally independent and only use a single model’s classification decision to categorize a data point. The aggregate judgment of these weak classifiers frequently results in a moderate, averaged output, thanks to ensemble training, which allows several weak classifiers to be combined into a family. It is reasonable to assume that each family of poor classifiers will make mistakes while using ensemble training. It is done in such a way that the correlation between each set of weak classifiers is kept to a minimum.

As a result, the likelihood of a certain mistake may not be as common in other groups as it is in one group. Therefore, there is a good possibility that any training technique that can effectively reduce these mistakes will increase the precision and robustness of machine learning algorithms. In order to reduce the mistakes from each model, ensemble training combines several decisions from many models. The algorithms previously mentioned are structurally independent and only use a single model’s classification decision to categorize a data point. The aggregate judgment of these weak classifiers frequently results in a moderate, averaged output, thanks to ensemble training, which allows several weak classifiers to be combined into a family. It is reasonable to assume that each family of poor classifiers will make mistakes while using ensemble training. It is done in such a way that the correlation between each set of weak classifiers is kept to a minimum.

As a result, the likelihood of a certain mistake may not be as common in other groups as it is in one group. Therefore, there is a good possibility that any training technique that can effectively reduce these mistakes will increase the precision and robustness of machine learning algorithms. In order to reduce the mistakes from each model, ensemble training combines several decisions from many models. The algorithms previously mentioned are structurally independent and only use a single model’s classification decision to categorize a data point. The aggregate judgment of these weak classifiers frequently results in a moderate, averaged output, thanks to ensemble training, which allows several weak classifiers to be combined into a family. It is reasonable to assume that each family of poor classifiers will make mistakes while using ensemble training. It is done in such a way that the correlation between each set of weak classifiers is kept to a minimum.

As a result, the likelihood of a certain mistake may not be as common in other groups as it is in one group. Therefore, there is a good possibility that any training technique that can effectively reduce these mistakes will increase the precision and robustness of machine learning algorithms. In order to reduce the mistakes from each model, ensemble training combines several decisions from many models. The algorithms previously mentioned are structurally independent and only use a single model’s classification decision to categorize a data point. The aggregate judgment of these weak classifiers frequently results in a moderate, averaged output, thanks to ensemble training, which allows several weak classifiers to be combined into a family. It is reasonable to assume that each family of poor classifiers will make mistakes while using ensemble training. It is done in such a way that the correlation between each set of weak classifiers is kept to a minimum.

As a result, the likelihood of a certain mistake may not be as common in other groups as it is in one group. Therefore, there is a good possibility that any training technique that can effectively reduce these mistakes will increase the precision and robustness of machine learning algorithms. In order to reduce the mistakes from each model, ensemble training combines several decisions from many models. The algorithms previously mentioned are structurally independent and only use a single model’s classification decision to categorize a data point. The aggregate judgment of these weak classifiers frequently results in a moderate, averaged output, thanks to ensemble training, which allows several weak classifiers to be combined into a family. It is reasonable to assume that each family of poor classifiers will make mistakes while using ensemble training. It is done in such a way that the correlation between each set of weak classifiers is kept to a minimum.

As a result, the likelihood of a certain mistake may not be as common in other groups as it is in one group. Therefore, there is a good possibility that any training technique that can effectively reduce these mistakes will increase the precision and robustness of machine learning algorithms. In order to reduce the mistakes from each model, ensemble training combines several decisions from many models. The algorithms previously mentioned are structurally independent and only use a single model’s classification decision to categorize a data point. The aggregate judgment of these weak classifiers frequently results in a moderate, averaged output, thanks to ensemble training, which allows several weak classifiers to be combined into a family. It is reasonable to assume that each family of poor classifiers will make mistakes while using ensemble training. It is done in such a way that the correlation between each set of weak classifiers is kept to a minimum.

As a result, the likelihood of a certain mistake may not be as common in other groups as it is in one group. Therefore, there is a good possibility that any training technique that can effectively reduce these mistakes will increase the precision and robustness of machine learning algorithms. In order to reduce the mistakes from each model, ensemble training combines several decisions from many models. The algorithms previously mentioned are structurally independent and only use a single model’s classification decision to categorize a data point. The aggregate judgment of these weak classifiers frequently results in a moderate, averaged output, thanks to ensemble training, which allows several weak classifiers to be combined into a family. It is reasonable to assume that each family of poor classifiers will make mistakes while using ensemble training. It is done in such a way that the correlation between each set of weak classifiers is kept to a minimum.

As a result, the likelihood of a certain mistake may not be as common in other groups as it is in one group. Therefore, there is a good possibility that any training technique that can effectively reduce these mistakes will increase the precision and robustness of machine learning algorithms. In order to reduce the mistakes from each model, ensemble training combines several decisions from many models. The algorithms previously mentioned are structurally independent and only use a single model’s classification decision to categorize a data point. The aggregate judgment of these weak classifiers frequently results in a moderate, averaged output, thanks to ensemble training, which allows several weak classifiers to be combined into a family. It is reasonable to assume that each family of poor classifiers will make mistakes while using ensemble training. It is done in such a way that the correlation between each set of weak classifiers is kept to a minimum.

As a result, the likelihood of a certain mistake may not be as common in other groups as it is in one group. Therefore, there is a good possibility that any training technique that can effectively reduce these mistakes will increase the precision and robustness of machine learning algorithms. In order to reduce the mistakes from each model, ensemble training combines several decisions from many models. The algorithms previously mentioned are structurally independent and only use a single model’s classification decision to categorize a data point. The aggregate judgment of these weak classifiers frequently results in a moderate, averaged output, thanks to ensemble training, which allows several weak classifiers to be combined into a family. It is reasonable to assume that each family of poor classifiers will make mistakes while using ensemble training. It is done in such a way that the correlation between each set of weak classifiers is kept to a minimum.

As a result, the likelihood of a certain mistake may not be as common in other groups as it is in one group. Therefore, there is a good possibility that any training technique that can effectively reduce these mistakes will increase the precision and robustness of machine learning algorithms. In order to reduce the mistakes from each model, ensemble training combines several decisions from many models. The algorithms previously mentioned are structurally independent and only use a single model’s classification decision to categorize a data point. The aggregate judgment of these weak classifiers frequently results in a moderate, averaged output, thanks to ensemble training, which allows several weak classifiers to be combined into a family. It is reasonable to assume that each family of poor classifiers will make mistakes while using ensemble training. It is done in such a way that the correlation between each set of weak classifiers is kept to a minimum.

As a result, the likelihood of a certain mistake may not be as common in other groups as it is in one group. Therefore, there is a good possibility that any training technique that can effectively reduce these mistakes will increase the precision and robustness of machine learning algorithms. In order to reduce the mistakes from each model, ensemble training combines several decisions from many models. The algorithms previously mentioned are structurally independent and only use a single model’s classification decision to categorize a data point. The aggregate judgment of these weak classifiers frequently results in a moderate, averaged output, thanks to ensemble training, which allows several weak classifiers to be combined into a family. It is reasonable to assume that each family of poor classifiers will make mistakes while using ensemble training. It is done in such a way that the correlation between each set of weak classifiers is kept to a minimum.
overfitting. Now, the loss function and optimization equation are used for updating the neural network. First, the model used binary cross-entropy loss as a loss function to calculate the difference between model prediction and the true result of each sample which is represented in (19).

\[
I = -\frac{1}{N} \sum_{i=1}^{N} y_i \cdot \log(\hat{y}_i) + (1 - y_i) \cdot \log(1 - \hat{y}_i) \tag{19}
\]

where \(N\) is the number of samples entered in the iteration, \(y_i\) is the truth value of sample \(i\), and \(\hat{y}_i\) is the predicted result of the model for sample \(i\).

For optimization of the model and updating the model weights based on data provided, ADAM optimizer [36] was used for the ANN as it has proved its efficiency and reliability recently with applications with small datasets as this research. The batch size used for the model was 43, and the number of epochs was 100 epochs, which was chosen empirically so that after this number the model reaches its steady state.

In conclusion, all the hyperparameters used for each model can be summarized in Table 2.

### IV. PERFORMANCE METRICS AND RESULTS

As the evaluation has high dimensionality, 6 models were used in training and predicting the needs of students for performance evaluation and provided correct help from their mouse dynamics behavior. Six different AI models are used for training and a fair comparison is performed between them to choose the best one, which gives the best result of correctly running the automatic performance evaluation system for evaluating the performed experimental steps while performing each experimental lab task scenario separately.

The AI Tranning set is collecting through the experiment design phase where the Instructor and Teacher Assistant will run the experiment by all possible correct experiment conduction methods. Afterwards, this dataset is used to train the AI algorithms with correct solutions. Regarding the complex circuits, the same idea is performed.

The performance metrics, which are used to compare the performance of 6 AI algorithms are the average weighted accuracy, Precision, Recall, F1-Score, and ROC Curve. Knowing that TP is the True Positive prediction for the model, FP is the False Positive prediction for the model, TN is the True Negative prediction for the model, FN is the False Negative prediction for the model.

First, the representation of the predictions has to be provided using the confusion matrix for all the tasks with each model. As can be observed in Fig. 9.a, Fig. 9.b, and Fig. 9.c, SVM has a fine confusion matrix with a concentration in diagonal which will be getting concentrated more by making more complicated tasks in the lab. This is happening due to the kernel added to SVM which is RBF. Also, after checking the results, the model is not underfitting as the result of the training and testing sample is too close.

NB was one of the worst ML models that can be used with such a problem. This has been observed based on Fig. 10.a, SVM confusion matrix for 3 tasks.

![Fig. 9. SVM confusion matrix for 3 tasks.](image)

| Model       | Hyper parameter | Value       |
|-------------|-----------------|-------------|
| SVM         | Kernel          | RBF         |
|             | C               | 5           |
| Naive Bayes | Type            | Gaussian    |
| KNN         | N-Neighbors     | 8           |
| Decision Tree | Max. Depth | 8           |
|             | Criterion       | Gini        |
| Random Forest | N-estimators | 1600        |
|             | Max. Depth      | 9           |
| ANN         | Epochs          | 100         |
|             | Optimizer       | ADAM        |
|             | Loss            | Binary Cross-Entropy |
Fig. 10.b, and Fig. 10.c, as most of the samples were predicted to be not correct for task 1, and the opposite side in tasks 2, and 3. This can be explained by understanding the nature of NB, which depends on conditional probability between class and feature, taking into consideration that this model treats every feature as independent from other features which makes it harder for the model to learn by building relations between features and each other.

As KNN is based on the concept of majority votes of nearby samples, it has made a quite progress in its results that is observed in Fig. 11.a, Fig. 11.b, and Fig. 11.c. Due to the similarity in the positive samples and diversity in the negative samples, most of the false predictions have been made on the False positive side (predicted correct but truly not correct).

According to Fig. 12.a, Fig. 12.b, and Fig. 12.c, the decision tree has evaluated the tests based on the tree in Fig. 7. Due to the concept of DF, the model is tending to overfit in such a task, which is not preferable lab evaluation problem due to the variety in the features and their intervals.

Using DF in such a problem may cause a long-term misclassification as the model will be related to the data trained on more than the problem concept itself. After evaluating the RF model using the test samples as shown in Fig. 13.a, Fig. 13.b, and Fig. 13.c, part of the DF disadvantage has been solved. As RF is based on the majority votes among various DTs, the model has achieved acceptable results disregarding the nature of the lab task itself. This can be observed in Fig. 13.b, and Fig. 13.c, as the result didn’t change or degraded, while the evaluated task became more complicated.

As noticed in Fig. 14.a, Fig. 14.b, and Fig. 14.c, ANN has promising confusion matrices. It is also observed that as the task gets complicated the model takes more advantage of the task. This can be demonstrated and explained due to the complicated architecture of ANN as its layers are trained to fit with the pattern of the problem. Hence, when the data is more complicated, the model has a more complicated pattern to learn which leads to better model predictions.
The Precision can be represented using (20), which is a classification model’s capacity to isolate just the pertinent data elements. Precision in mathematics is the product of the number of true positives and the sum of the true positives and
false positives.

\[
\text{Precision} = \frac{TP}{TP + FP} \quad (20)
\]

The Recall can be calculated using (21), which is the capacity of a model to locate each and every pertinent example in a data collection. The number of true positives is divided by the sum of the true positives and false negatives.

\[
\text{Recall} = \frac{TP}{TP + FN} \quad (21)
\]

The F1-Score can be reached using precision and recall as in (22), which is the average of Precision and Recall, weighted. Therefore, both false positives and false negatives are included while calculating this score. Although F1 is typically more valuable than accuracy, especially if you have an uneven class distribution, it is not intuitively as simple to grasp as accuracy.

\[
F1 - \text{Score} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \quad (22)
\]

The accuracy can be calculated using (23), which is the easiest to understand performance metric, as it simply measures the proportion of properly predicted observations to all observations.

\[
\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \quad (23)
\]

The models with the best performance based on accuracy score are bolded to be clear for observation.

Regarding ROC Curve, it is plotted using a False Positive Rate (FPR), which can be calculated using (24), True Positive Rate (TPR) (which is already defined as Recall).

\[
\text{FPR} = \frac{FP}{FP + TN} \quad (24)
\]

As appears in Table 3, Table 4, and Table 5, the best overall performance can be reached with the ANN. Also, it is noticed that the model can have a better performance as the task is getting more complicated. As all the models had better overall performance in task 3 than in task 2 or task 1. Regarding RF and SVM, they had close performance to each other. This can be due to their nature in dealing with high-dimensional features, as RF deals with numerous numbers of decision trees to have majority votes, and SVM uses the kernel to make it easier for the model to have a hyperplane.

### TABLE 3. Models performance metrics for task 1.

| Model         | Accuracy | Precision | Recall | F1-Score |
|---------------|----------|-----------|--------|----------|
| SVM           | 0.7316   | 0.73      | 0.76   | 0.74     |
| Naive Bayes   | 0.4895   | 0.50      | 0.06   | 0.11     |
| KNN           | 0.7368   | 0.78      | 0.67   | 0.72     |
| Decision Tree | 0.6474   | 0.67      | 0.62   | 0.64     |
| Random Forest | 0.7789   | 0.82      | 0.73   | 0.77     |
| ANN           | 0.7947   | 0.80      | 0.80   | 0.80     |

### TABLE 4. Models performance metrics for task 2.

| Model         | Accuracy | Precision | Recall | F1-Score |
|---------------|----------|-----------|--------|----------|
| SVM           | 0.8485   | 0.85      | 0.85   | 0.85     |
| Naive Bayes   | 0.6418   | 0.65      | 0.91   | 0.74     |
| KNN           | 0.7929   | 0.77      | 0.84   | 0.80     |
| Decision Tree | 0.8081   | 0.80      | 0.82   | 0.81     |
| Random Forest | 0.8586   | 0.87      | 0.85   | 0.86     |
| ANN           | 0.8788   | 0.91      | 0.84   | 0.87     |

### TABLE 5. Models performance metrics for task 3.

| Model         | Accuracy | Precision | Recall | F1-Score |
|---------------|----------|-----------|--------|----------|
| SVM           | 0.8767   | 0.85      | 0.91   | 0.88     |
| Naive Bayes   | 0.7306   | 0.67      | 0.91   | 0.77     |
| KNN           | 0.8402   | 0.78      | 0.94   | 0.85     |
| Decision Tree | 0.7580   | 0.72      | 0.85   | 0.78     |
| Random Forest | 0.8767   | 0.86      | 0.90   | 0.88     |
| ANN           | 0.8858   | 0.90      | 0.86   | 0.88     |

Regarding KNN, it has acceptable performance, although it is simple. It has also appeared that DT can’t have a stable performance among all tasks, which makes it not reliable all the time. This may lead to difficulty in expecting its performance without having multiple analyzes in various similar tasks. Unfortunately, NB doesn’t perform well in all tasks which leads to disqualifying it from any further research in the field of mouse dynamics.

More illustrated results can be understood in Fig. 15, Fig. 16, and Fig. 17 using ROC Curve. To be more clear, as the peak of the model graph tends to the top left, the performance is going better, which means it has a higher Area Under the Curve (AUC). In the ROC curve of task 1 (Fig. 15), all models have near results except NB, which had poor performance. ANN and RF had close results in this task. In the ROC curve of task 2 (Fig. 16), all models have close TPR but differ in FPR. This causes the upper hand for ANN in task 2. Finally, in the ROC curve of task 3 (Fig. 17), SVM and RF have the same performance, which comes after ANN. Despite most of the models having better TPR, the cause of ANN leading...
in the best performance in all tasks came from lowering the FPR.

From previous results, it is noticed that the Noval ANN has the best performance with 79.47% in task 1, 87.88% in task 2, and 88.58% in task 3, respectively. Coming after it RF and SVM respectively with 77.89% - 73.19% in task 1, 85.8% - 84.85% in task 2, and 87.67% - 87.67% in task 3. And it is noticed throughout the whole study that as the task is developed to be more complicated, the performance rises for all the models, and this can be demonstrated that having a complex task to evaluate, makes it easier for the model to get its pattern reasonably. It is preferable to disqualify NB in any research regarding mouse dynamics in the future. Also, through the study, it is noticed that balanced data is a need to reach acceptable results that can be reliable for the problem.

V. EXPERIMENT SURVEY STUDY

An experiment survey study has been conducted in Helwan University among 25 students to test the test version of the system and fill out a survey form regarding the reliability, applicability, and usability of the software. The questions of survey were as follows:

1. In a scale of 1 to 10, rate the applicability of the project.
2. In a scale of 1 to 10, rate the usability of the project.
3. In a scale of 1 to 10, Do you think this software can help students who are struggling in practical lab tasks?
4. In a scale of 1 to 10, Do you think this software will be reliable to grade the practical lab tasks for students?
5. In a scale of 1 to 10, Do you think this software will evolve to assist the Staff in the learning process?
6. Choose the best 3 advantages of this system.
   i. Help students with learning in case of educational staff shortage
   ii. Contribute in the efficiency of the learning process in general
   iii. Improve the e-learning process
   iv. Provide a generalized system to be applied to any software simulation
   v. The lack of hardware dependency due to models simplicity
to another, and depending on the evaluation method, it may need high computational power. In this research, a method for automated performance evaluation for the experiments is proposed to provide an assistant for the students while performing the experiments. Furthermore, to evaluate the experimental steps performed by students to provide auto-grading for different types of questions solved by students in practical exams. As a consequence, the new performance evaluation module performs both the lab evaluation and assistance automatically, starting from the phase of fetching data, then feature extraction using statistical approaches. After that, features preprocessing to avoid overfitting and dropping unneeded features, then reaching the stage of implementing various ML models to use for evaluation and compare them trying to declare the best model for the problem. As the evaluation has high dimensionality, 6 models were used in training and predicting the needs of students for performance evaluation and a number of metrics are collected for three different tasks. It was found that the Artificial Neural Network (ANN) is the best AI algorithm that can be used in the performance evaluation module, with an average approximated precision value of 87% for three tasks. Furthermore, it achieved the highest average approximated accuracy and F1-score of 85% and 85%, respectively for the three tasks. Followed the Support Vector Machine (SVM), with an average approximated value of precision, accuracy, and F1-score equals 81%, 81%, and 82%, respectively for 3 tasks.

The new virtual service method is integrated within a completely new LLMS. This system provides the students and teachers with the flexibility of running any software within experiments remotely using a single interface. Also, the new LLMS provides an automatic performance evaluation system that studies the students’ mouse behavior while performing experiments or exams remotely. This new performance evaluation system evaluates if the students performed the lab steps correctly or not to help the auto-grading module to provide the correct grade to help the teacher in evaluating all types of experiments or exams. Furthermore, the new performance evaluation system triggers the Virtual Assistant Service that is used to provide the students with needed help and run one of the Session Collaborator’s features through the Experimental Runtime Engine Module of the LLMS system. Furthermore, the new LLMS system provided the student with the ability to control the laboratory’s equipment physically exists in the laboratory remotely.

REFERENCES

[1] M. H. C. Lin Chen and K. S. Liu, “A study of the effects of digital learning on learning motivation and learning outcome,” Eurasia J. Math., Sci. Technol. Educ., vol. 13, no. 7, pp. 3553–3564, 2017.

[2] H. M. S. Ahmed, “Hybrid E-learning acceptance model: Learner perceptions,” Decis. Sci. J. Innov. Educ., vol. 8, no. 2, pp. 313–346, Jul. 2010.

[3] I. Lykourentzou, I. Gianougkos, G. Mpardis, V. Nikolopoulos, and V. Loumos, “Early and dynamic student achievement prediction in E-learning courses using neural networks,” J. Amer. Soc. Inf. Sci. Technol., vol. 60, no. 2, pp. 372–380, Feb. 2009.
[4] A. Baki, L. Oubahssi, C. Cherkauoi, and S. George, “Motivation and engagement in MOOCs: How to increase learning motivation by adapting pedagogical scenarios?” in *Design for Teaching and Learning in a Networked World*. Cham, Switzerland: Springer, 2015, pp. 556–559, doi: 10.1007/978-3-319-24258-3_58.

[5] M. Vahdat, L. Oneto, D. Anguita, M. Funk, and M. Rauterberg, “A learning analytics approach to correlate the academic achievements of students with interaction data from an educational simulator,” in *Design for Teaching and Learning in a Networked World*. Cham, Switzerland: Springer, 2015, pp. 352–366, doi: 10.1007/978-3-319-24258-3_56.

[6] N. Tomasevic, N. Gvozdenovic, and S. Vranes, “An overview and comparison of supervised data mining techniques for student exam performance prediction,” *Comput. Educ.*, vol. 143, Jan. 2020, Art. no. 103676.

[7] A. Hellas, P. Iantola, A. Petersen, V. V. Ajanovski, M. Gutica, T. Hynynen, A. Knutas, J. Leinonen, C. Messom, and S. N. Liao, “Predicting academic performance: A systematic literature review,” in *Proc. Companion 23rd Annu. ACM Conf. Innov. Technol. Comput. Sci. Educ.*, 2018, pp. 175–199.

[8] D. Bueno-Fernández, D. Gil, and S. Luján-Mora, “Application of machine learning in predicting performance for computer engineering students: A case study,” *Sustainability*, vol. 11, no. 10, p. 2833, May 2019.

[9] S. Huang and N. Fang, “Predicting student academic performance in an engineering dynamics course: A comparison of four types of predictive mathematical models,” *Comput. Educ.*, vol. 61, pp. 133–145, Feb 2013.

[10] H. Wei, H. Li, M. Xia, Y. Wang, and H. Qu, “Predicting student performance in interactive online question pools using mouse interaction features,” in *Proc. 10th Int. Conf. Learn. Analytics Knowl.*, Mar. 2020, pp. 645–654.

[11] S. Liu and M. d’Aquin, “Unsupervised learning for understanding student achievement in a distance learning setting,” in *Proc. IEEE Global Eng. Educ. Conf. (EDUCON)*, Apr. 2017, pp. 1373–1377.

[12] G. B. Brahim, “Predicting student performance from online engagement activities using novel statistical features,” *Arabian J. Sci. Eng.*, vol. 47, no. 8, pp. 10225–10243, Aug. 2022.

[13] T. Hu, W. Niu, X. Zhang, X. Liu, J. Lu, and Y. Liu, “An insider threat detection approach based on mouse dynamics and deep learning,” *Secur. Commun. Netw.*, vol. 2019, pp. 1–12, Feb. 2019.

[14] Á. Fülöp, L. Kovács, T. Kurics, and E. Windhager-Pokol. (2016). Balabit Mouse Dynamics Challenge Dataset. [Online]. Available: https://github.com/balabit/Mouse-Dynamics-Challenge

[15] M. Antal and N. Fejér, “Mouse dynamics based user recognition using deep learning,” *Acta Universitatis Sapientiae, Inf.*, vol. 12, no. 1, pp. 39–50, Jul. 2020.

[16] M. Antal and L. Denes-Fazakas, “User verification based on mouse dynamics,” *Proc. 10th Int. Conf. Learn. Analytics Knowl.*, vol. 15, no. 1, pp. 1929–1958, May 2019.

[17] H. Wei, H. Li, M. Xia, Y. Wang, and H. Qu, “Predicting student performance in interactive online question pools using mouse interaction features,” in *Proc. 10th Int. Conf. Learn. Analytics Knowl.*, Mar. 2020, pp. 645–654.

[18] S. Liu and M. d’Aquin, “Unsupervised learning for understanding student achievement in a distance learning setting,” in *Proc. IEEE Global Eng. Educ. Conf. (EDUCON)*, Apr. 2017, pp. 1373–1377.

[19] G. B. Brahim, “Predicting student performance from online engagement activities using novel statistical features,” *Arabian J. Sci. Eng.*, vol. 47, no. 8, pp. 10225–10243, Aug. 2022.

[20] T. Hu, W. Niu, X. Zhang, X. Liu, J. Lu, and Y. Liu, “An insider threat detection approach based on mouse dynamics and deep learning,” *Secur. Commun. Netw.*, vol. 2019, pp. 1–12, Feb. 2019.

[21] A. Fülöp, L. Kovács, T. Kurics, and E. Windhager-Pokol. (2016). Balabit Mouse Dynamics Challenge Dataset. [Online]. Available: https://github.com/balabit/Mouse-Dynamics-Challenge

[22] M. Antal and L. Denes-Fazakas, “User verification based on mouse dynamics,” *Proc. 10th Int. Conf. Learn. Analytics Knowl.*, vol. 15, no. 1, pp. 1929–1958, May 2019.

[23] M. Antal and L. Denes-Fazakas, “User verification based on mouse dynamics,” *Proc. 10th Int. Conf. Learn. Analytics Knowl.*, vol. 15, no. 1, pp. 1929–1958, May 2019.

[24] M. A. A. El-Haleem et al.: Generic AI-Based Technique for Assessing Student Performance

[25] H. Gamboa and A. Fred, “A behavioral biometric system based on human-computer interaction,” in *Proc. SPIE*, 2004, pp. 381–392.

[26] M. Cogswell, F. Ahmed, R. Girshick, L. Zitnick, and D. Batra, “Reducing overfitting in deep networks by decorrelating representations,” in *Proc. ICLR*, 2016, pp. 1–12.

[27] P. Rodríguez, J. González, G. Cucurull, J. M. Fonfau, and X. Roca, “Regularizing CNNs with locally constrained decorrelations,” in *Proc. ICLR*, Mar. 2017, pp. 1–11.

[28] N. Cristianini and J. Shawe-Taylor, *An Introduction to Support Vector Machines and Other Kernel-Based Learning Methods*. Cambridge, U.K.: Cambridge Univ. Press, 2000.

[29] V. N. Vapnik, *The Nature of Statistical Learning Theory*. New York, NY, USA: Springer, 1995.

[30] H. Reza, “Eye tracking with EEG life-style,” Ph.D. thesis, Dept. Elect. Eng., Birmingham Univ., Birmingham, U.K., 2015.

[31] G. Guo, H. Wang, D. Bell, Y. Bi, and K. Greer, “KNN model-based approach in classification,” in *On The Move to Meaningful Internet Systems 2003: CoopIS, DOA, and ODBASE*. Berlin, Germany: Springer, 2003, pp. 986–996, doi: 10.1007/978-3-540-39964-3_62.

[32] Q. Ren, M. Li, and S. Han, “Tectonic discrimination of olivine in basalt using data mining techniques based on major elements: A comparative study from multiple perspectives,” *Big Earth Data*, vol. 3, no. 1, pp. 8–25, Jan. 2019.

[33] A. Zhandos and J. Guo, “An approach based on decision tree for analysis of behavior with combined cycle power plant,” in *Proc. Int. Conf. Prog. Informat. Comput. (PIC)*, Dec. 2017, pp. 415–419.

[34] N. Siddiqui, R. Dave, M. Vanamala, and N. Seliya, “Machine and deep learning applications to mouse dynamics for continuous user authentication,” *Mach. Learn. Knowl. Extraction*, vol. 4, no. 2, pp. 502–518, May 2022.

[35] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, “Dropout: A simple way to prevent neural networks from overfitting,” *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 1929–1958, 2014.

[36] D. Kingma and J. Ba, “Adam: A method for stochastic optimization,” in *Proc. 3rd Int. Conf. Learn. Represent.*, 2014, pp. 1412–6980.

AHMED M. ABD EL-HALEEM received the B.Sc., M.Sc., and Ph.D. degrees in electronics and communications engineering from Helwan University, Cairo, Egypt, in 2001, 2006, and 2012, respectively. He is currently an Associate Professor with the Electronics and Communications Engineering Department, Faculty of Engineering, Helwan University. His current research interests include mobile/vehicular ad-hoc communication networks, 5G and 6G radio access networks, cognitive radio networking, device to device communication, the Internet of Things (IoT), reconfigurable intelligent surface (RIS), and AI application in wireless communication. This includes mobility management techniques and routing schemes for mobile ad-hoc networks (MANET), security and secure routing algorithms for ad-hoc wireless networks, and routing protocols for cognitive radio communication networks. This includes AI-based applications for education as well as wireless networks, the Internet of Things (IoT) communications, unmanned aerial vehicle (UAV) communications, reconfigurable intelligent surface (RIS), and smart Grid Communication systems. He is a member of the Research Team that awards several applied research projects funded by National and International funding agencies in the field of wireless communication, the IoT and its applications, and smart education systems.
MOHAB MOHAMED EID is currently pursuing the B.Sc. degree with the School of Computer Engineering, Nile University. He is currently working as a part-time Research and Development Machine Learning Engineer with Helwan University, and a Junior Teaching Assistant with the School of Engineering, Nile University. His research interests include autonomous vehicle, federated learning, autoML, machine learning into production, image processing, convolutional neural networks (CNN) optimization, sequential models, feature extraction, and filtration.

MAHMOUD M. ELMESALAWY (Member, IEEE) was born in Cairo, Egypt, in 1981. He received the B.S., M.S., and Ph.D. degrees in electronics and communications engineering from Helwan University, Cairo, in 2002, 2005, and 2010, respectively, the master’s degree in educational planning and management from IIEP, UNESCO, France, in 2012, and the micromaster’s degree in managing technology and innovation from RWTH Aachen University in collaboration with edX, Germany, in 2020. He is currently working as a Professor of wireless communications with the Department of Electronics and Communications Engineering and the Dean of the Faculty of Engineering, Helwan University, Egypt. He is also working as the Head of the Strategic Planning Unit with Helwan University. He has participated in and led several research and development projects funded by various international and national agencies. His current research interests include B5G and 6G mobile communication technologies and its applications in different fields such as healthcare and education. This includes AI-based applications for healthcare and education as well as wireless networks, the Internet of Things (IoT) communications, unmanned aerial vehicle (UAV) communications, reconfigurable intelligent surface (RIS), and smart grid communication systems.

HADEER A. HASSAN HOSNY received the M.Sc. and Ph.D. degrees in computer engineering from the Faculty of Engineering, Helwan University, in 2016 and 2021, respectively. She is currently an Assistant Professor of computer and systems engineering with the Faculty of Engineering, Helwan University. She is involved in many research projects. Her research interests include e-learning, artificial intelligence, energy-aware scheduling, scheduling of operating systems, reliability-aware scheduling, cloud computing, and parallel processing.