Human-Robot Interaction: Student Engagement Detection

Swansea University
Prifysgol Abertawe

Noah Isaac Roberts

975804

Department of Computer Science
Swansea University

Project Dissertation submitted to Swansea University in Partial Fulfilment for the Degree of Master of Science

September 2022

Declaration

This work has not been previously accepted in substance for any degree and is not being con- currently submitted in candidature for any degree.

Statement 1

This thesis is the result of my own investigations, except where otherwise stated. Other sources are acknowledged by footnotes giving explicit references. A bibliography is appended.

Statement 2

I hereby give my consent for my thesis, if accepted, to be made available for photocopying and inter-library loan, and for the title and summary to be made available to outside organisations.

Abstract

In this thesis we explore human-robot interaction, more specifically the engagement of a student whilst interacting with a robot. A system has been implemented to discover a student's engagement during a video. Engagement has been deduced by the student's facial expression, gaze direction, number of blinks, number of poses, and lastly, number of words spoken during the video. Our system has been used on many videos featuring different students to create a vast data set. Finally, machine learning techniques are used to analyse the data set and discover the accuracy of our system. Using our system, we were able to detect 3 different levels (low, medium, high) of engagement with up to 89% accuracy.

Table of Contents

Introduction

Motivation

Student engagement has been an interesting topic in academic literature of recent times with many struggling with the literal definition and if it can be generally defined. It is an important topic with the classroom becoming more advanced and robots being used in education more commonly. By detecting student engagement, we can find out what strategies work best for learning.

This project aims to further build on current research of human-robot interaction but more specifically detecting a student's engagement whilst interacting with a robot (tutor bot). We want to find out if robots can engage students, this is a very important "if" as in 2016 the UNESCO Institute for Statistics estimated that by 2030 the world would need 69 million new teachers and tutors "to provide quality universal primary and secondary education" (Degree Choice., 2021). If robots can replace teachers in a classroom a very big demand for teachers and tutors in the education sector can be filled. This project implements a system to detect student engagement from a video, whilst interacting with a robot. Using our system, a dataset has been created and machine learning techniques have been performed to discover the accuracy of our system. During the related work chapter, we dive into different literature to ensure that for our project we had a concrete understanding of what indicates engagement from a student, including components such as behavioural to cognitive.

We found multiple problems in this field of research. As previously mentioned, student engagement struggles to be defined, have we further helped clarify this definition? Can student engagement be accurately detected? Tutor bots are a new technology within education, can they be used to aid teachers or even replace them? The motivation behind this project came from these questions and we aim to answer them within this thesis.

Another problem with current education is students not learning optimally. By detecting engagement, we can spot what keeps students engaged and if robots can

improve this. (Schnitzler et al., 2021) reviews the improvement of student academic achievement when more engaged with their learning, this is more thoroughly discussed in the following chapter. This shows further improvements can be made in education to ensure optimal engagement from students and motivates this project to have furthered the research in this area. As demand for tutors increases educators are looking for innovative ways to fix this worldwide issue. With the use of technology in the education constantly evolving an opportunity has arisen for robots but more specially tutor-bots to help satisfy this demand. The use of robots in education has been rare due to the economic constraints they originally had, however in more recent times their cost and performance have vastly improved (Merkle., 2003). Robots are now mass-produced, affordable and trusted to be used in the classroom and for personal tutoring needs, an example of this being the Lego Mindstorms which allows students to follow design processes to solve different challenges. When using robots as tutor bots we want to make sure that students are engaged whilst learning. With the results achieved from this project we can identify if this is the case.

Aims and Objectives

In this thesis we discuss the implementation of our student engagement system. The aim of the project is to be able to detect student engagement whilst interacting with a tutor bot. During the related work chapter, current solutions and their limitations have been discussed. We aim to discover through research how students engage when learning and how they engage with tutor bots so that we can solve the problems mentioned in the previous subchapter. This has helped us explore and answer the following question:

1. What level of engagement does a student show when interacting with a tutor bot?
2. What attributes suggest a student is engaged and can we detect it accurately?
3. Can we help define student engagement?

During the results section we will discuss the accuracy of our system by comparing the results of different ML techniques that were used on our dataset.

The technical aims of this project:

1. A software solution that detects student engagement from a video has been implemented
2. A dataset has been created by using the solution on different videos
3. Machine learning techniques have been performed on the dataset to discover the accuracy of our solution

Thesis Overview

This thesis is split up into the following chapters:

- Introducing the project, its aims and motivation
- Presenting background research and similar applications in this area
- Discussing the technological choices used in the project
- Planning of the project and time management
- Discussing the design and implementation of the project
- Discussing future work
- Concluding the thesis
- Presenting all references for the thesis
- Appendix presenting images or code snippets not included in the main body of the thesis

Background Research

This research and partial literature review will first follow a thematic approach where sub-topics will be separated and discussed independently. The subtopics for this review will include robots in education, and engagement – with multiple engagement subchapters. We will also discuss these areas combined and what solutions have been previously implemented. This will help highlight the problem and how our solution aims to fix it.

Robots in Education

(Mublin et al., 2013) reviewed the use of robots being utilised as an educational technology. The reviews criteria included "domain of the learning activity, location of the activity, the role of the robot, types of robots and types of robotic behaviour". They discovered that the primary use for robots was to provide language, science, or technology education in the form of a "tutor, tool or peer in a learning activity". The paper serves as a useful guide to robots in education, further development into using robots as a peer could provide key insight into human-robot interaction.

(Toh et al., 2014) conducted a review of robots in education and young children. The paper synthesised findings of research studies carried out in the previous 10 years to analyse the influence of robots on children's learning. They grouped the robots influence on children's development into four major categories: cognitive, conceptual, language and social skills. They discovered that from a parent's perception results were mixed when it came to the use of robots in their children's education. This may lead others to think that robots do not have a strong place in education, however further research must be done in the area and is a strong motivator for this project.

(Belpaeme et al., 2018) conducted a review into the use of social robots as tutors or peer learners and found that they can increase cognitive and affective outcomes. They also review the potential of social robots in education and consider "how the robot's appearance and behaviour affect learning outcomes" (Belpaeme et

al., 2018). When analysing the benefits of social robots as tutoring agents they found that physical robots were more likely to enhance the learning for a student as they were more engaging and enjoyable than a virtual tutoring agent. They discovered that robots with a like human presence resulted in the most positive learning outcomes, an example of this being the Nao robot – a humanoid that could walk, make gestures, and move its head. During the research they found robots in education to be limited by technical and logistical challenges however "the benefits of physical embodiment may lift robots above competing learning technologies" (Belpaeme et al., 2018).

(Konjin et al., 2020) reviewed the use of robots in education. They categorised the types of robots in education, build robots, use robots and social robots (Catlin et al., 2018). The first two are mainly used as tools for constructing robots and learning how to program, the type were interested in is the social robot. Social robots deliver a learning experience through interaction with students (Konjin et al., 2020). They are used in place of a teacher and are used as a "tutor bot". This review provided further motivation during this project; they summarised that social robots have greater advantages over screen-based technology. They discuss the importance of investigating what features on a human-like robot improve the student's learning experience. During our project we aimed to discover this by detecting behaviours a student exhibits whilst optimally engaged or disengaged with a tutor bot.

Defining Engagement

In this subsection we will be discussing the research that aims to define engagement. Researching engagement gave a greater understanding of what we needed to detect using our system to indicate if the student is "engaged" and at what level.

(Kelders et al., 2020) investigated the components of engagement within an eHealth context. They aimed to get a clear understanding of engagement to help development of eHealth technologies. They discovered that engagement is "predominantly conceptualised as a multidimensional. construct with three common components" (Kelders et al., 2020). These components being behaviour, cognitions, and affective. They summarised 18 studies classified as student engagement, 11 of

these studies mentioned behaviour, emotion, and cognition as components of engagement.

(Schnitzler., 2021) researched student engagement patterns and their relation to academic achievement. The paper discusses a survey of a multiple classroom surveys where hand-raising interacts along with student cognitive, and emotion were used as engagement indicators. Engagement patterns were split into 5 categories: engaged, disengaged silent compliant and busy. The paper summarised that the students with greater academic success had been more engaged during the classroom sessions than the students with less academic success who showed low to no level of engagement. However, the paper suggests further differentiation is needed between engagement and disengagement as they said, "being engaged can take on various forms, from compliant to busy" (Schnitzler., 2021). This suggest that although a student may be compliant with a task they are not necessarily engaged, and that their learning is not optimised.

Although some studies suggest further differentiation into the definition of student engagement, a common theme of cognitive, behavioural, and emotional components being used to indicate student engagement helped influence our decision in what to detect within our own solution. The literature in this field indicated that these components would lead to the most accurate results.

Engagement Behaviours and Detection

This subsection discusses current literature for detecting engagement, what behaviours indicate engagement and solutions that have been implemented and their limitations.

(Sharma et al., 2019) presented a system that can detect the level of engagement in a student. They used a built-in webcam from a typical laptop which would analyse the student in real time. Movement of the head and eyes was combined with facial expressions to produce a "concentration index with three classes of engagement: "very engaged", normally engaged" and "not engaged at all".

They tested the system with an e-learning scenario and were able to correctly identify a student's level of engagement. They discovered that students with the highest scores had a higher concertation index. The paper lacks analysis on different learning strategies, further developments in this domain would help discover maximum engagement from a student.

(Ali et al., 2019) conducted a review of engagement in online learning, they put on various educational activities that included reading, writing, video tutorials, online exams, and online meetings. They would monitor different engagement levels show by the students such as "boredom, frustration, delight, neutral, confusion, and learning gain" (Ali et al., 2019). Using the feedback from student engagement they were able to provide personalised pedagogical support to online learners. They examined the use of facial expression in greater detail as this found the most promising results in terms of student engagement compared to other means of monitoring such as audio and text. As a result of their results, they were able to provide recommendations for the future to "advance the technology of engagement detection for online students" (Ali et al., 2019). The paper lacks in using interactivity to improve student engagement, further development into this area would help develop the students learning experience.

This section has helped identify behaviours that indicate engagement and how to detect them, this was key for our project and helped contribute towards choosing the following to be detected for engagement in our own project:
Emotion, gaze, number of blinks, number of poses, and number of words spoken.

Emotion Detection

Jaiswal et al., 2019) discuss a model they implemented to capture emotion in real time using a CNN architecture. They mention the need for automation in every field with the increase of human-robot interaction in current times, also the need to understand human emotion. Their model predicts emotion of an image using a CNN with reduced parameters by "90x from that of Vanilla CNN and also 50x from latest state of the art research" (Jaiswal et al., 2019). They were able to record accuracies of

74% with their system. Although emotion shows some indication of engagement there are a lot more deciding factors which has led us to explore other components as well as emotion in our own system.

(Sun et al., 2004) explores trends in emotional intelligence in HCI paradigms. In their paper they discuss the facial expression database they created and evaluate the emotion detection with different learning ML techniques including Bayesian Networks, SVMs, and Decision trees. They found highest accuracies with a CNN (4.43, 0.97%) but found disadvantages with this such as slow computation (Sun et al., 2004). Our own system uses OpenCV's Haar Cascade Classifier which uses CNN by default.

Gaze Tracking

(Ohno et al., 2002) introduced a gaze tracking system, "FreeGaze". The aim was to create a gaze detection system for everyday gaze interaction. They use a geometric eyeball model with sophisticated image. Processing. Their system only needed two points for calibration and boasted of their uncomplicated system. However, with eye gaze technology greatly improving in recent years, this solution became quickly outdated and unreliable compared to newer systems.

Some gaze tracking systems can be intrusive and unnatural, (Stiefelhagen et al., 1997) presented a non-intrusive model-based gaze tracking system. Their system estimates the head of a user using six facial feature points, and tracks features such as eyes, nostrils, and the corner of lips. They used a full perspective model to map feature points and several techniques are used to track gaze. Again, this model became outdated with current models using hundreds of landmarks to detect features and track gaze much more accurately. Newer models will now be discussed.

(Lame et al., 2020) created Gaze Tracking, an eye tracking system. The library is easily implementable within Python projects and was used for gaze detection during this project. The system gives the exact position of pupils and gaze in real-time by using a shape predictor model with 68 facial landmarks and OpenCV functions for finding the centre of the eyeball. This is done by storing the x and y values of both eyes from the eye shape landmarks and draw them on a black mask using a function “fillConvexPoly” (Agarwal., 2020). With knowledge of the area between these points the eye area can be drawn in white on the black mask. The white area is expanded with the dilate function and pixels are converted. From (0,0,0) to (255,255,255) so that eyeball is only dark area (Agarwal., 2020), see example in figure 1 of process. This library was the most accurate system for the type of gaze detection that we needed for this project. As mentioned, its ease of use and implementation with existing project also led

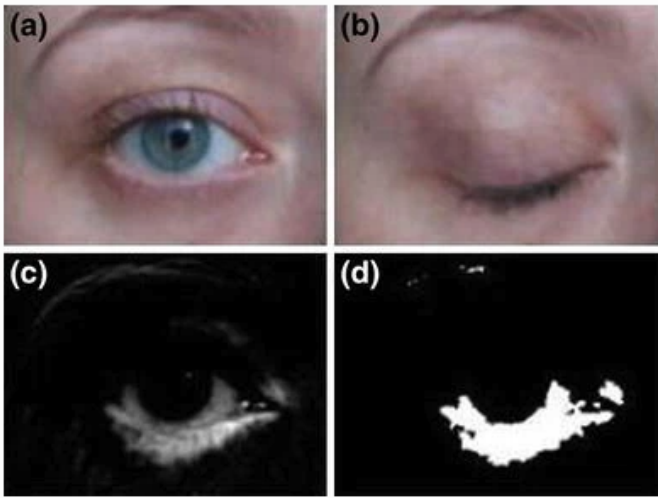


Figure 1: Eye detection method

Although gaze detection can be unreliable due to lighting issues or distance from. camera, the decision to use it was still made as the videos being used in our experiment were recorded in a controlled environment without disruption. We decided it to be a very important factor when determining engagement as in our experiment the student shows signs of engagement when looking in only two directions, centre (at the robot) or down (at the tablet). Due to the undisturbed environment and controlled lighting the videos were taking in, the chances of unreliable gaze tracking were low.

Blink Detection

(Krolak et al., 2012) presents an eye-blink detection system for human-computer interaction. Their method involves detecting voluntary eye-blinks and the use image processing methods (Haar) for face detection, and "template matching based eye tracking and eye-blink detection" (Krolak et al., 2012). The method involves a similar process as discussed in the gaze tracking section, with an example shown in figure 1. Disadvantages of these systems are that eyes cannot always be tracked e.g., when student is looking down with their head tilted. To combat these disadvantages our system a system for looking down has been implemented. This was extremely important for our system as looking down means that the student is engaged with the tablet suggesting although gaze/blinks cannot be tracked, the student is engaged.

(Morris et al., 2002) discussed their blink detection system with the goal of providing non-contact head and eye controls for people with motor difficulty. Their system makes use of spatiotemporal filtering and variance maps to locate the head and eye points. They were able to track these feature points accurately by making use of a modified Lucas-Kanade tracking algorithm. They summarised their results. And found that accurate results were obtained at a processing rate of more than 30 fps in greater than 90% of cases. They were able to achieve a false positive blink detection rate of 0.01% (Morris et al., 2002). The paper presented impressive results for the time it was wrote and gave excellent insight into blink detection systems.

(Lame et al., 2020) gaze detection library that we chose to use for gaze detection during this project also comes with a blink function to detect blinks. The blinking ratio is calculated by dividing the width of the eye by its height. The function takes in the parameter for the facial landmarks and the points of an eye (Lame et al., 2020). An issue arose using this function of too many blinks being counted, to solve this a Boolean check was put in place to not count a blink until the eye had closed and fully opened again. This way of detecting blinks was the obvious choice with having already have used the library for gaze detection. Having monitored the blink detection in an earlier prototype of the system we found the count to be very accurate and therefore were happy with our choice.

Pose Detection

(Rogez et al., 2008) presents a pose detection system from videos by "formulating it as a classification problem". They align training images using 3D Mocap data and define classes using a 2D manifold: camera viewpoint and actions. They describe the process of their model which is based on random forests. A bottom-up approach is followed to build a list of features using comparison with the training images (Rogez et al., 2008). They were able to achieve promising results with both fixed and moving cameras. Although good work in the field of pose detection, this paper only discusses the use of one ML technique. To improve a comparison with multiple techniques may bring more accurate results. Our thesis builds on work such as this by using multiple techniques.

(Obdrzalek et al., 2012) present a real-time algorithm for pose detection. They used vision-based 3D data and tele-rehabilitation in virtual environments, cameras capture movements in real-time and send data remotely. Their pose detection algorithm "extracts kinematic parameters for participants and determines pose similarity" (Obdrzalek et al., 2012). They also provide real-time feedback, providing a pose similarity score. The system shows great use of an algorithm for pose detection that motivated the potential use of an algorithm in our own work.

Although (Obdrzalek et al., 2012) produced accurate results with an algorithm-based pose detection system we chose against using such a model. Their system requires multiple stereo cameras for data capture. Our own system aimed for ease of use, making use of a laptops built in features (camera and mic) for every aspect of engagement detection. This led us to look for alternatives, eventually going back to the Mediapipe library previously used for aspects of face detection. They offer full body pose detection as well as individual parts of the body. As use of our system features a child sat behind a desk our aim was to detect hand movement for gestures and poses. The library gives a hand and finger tracking solution easily implementable in projects (see implementation for further technical discussion). Like the blink counter, a Boolean check was used to see if a hand had appeared on-screen – this would count as a pose. This method of pose detection worked well as we didn't

need to go further in depth of algorithm use and implementation for a small section of this project.

Speech Detection

A speech-to-text method of speech detection was used in this project. The speech_recognition library was used for this and has been discussed in the implementation chapter. A brief overview of how the library works is that it contains a recogniser class that makes use of 7 APIs to check a word appears within word databases. As we were counting the number of words spoken by a student in a video this method worked well, however before picking this method we investigated other ways to detect speech.

(Shafran et al., 2003) provides a solution for robust speech detection. Their solution uses an algorithm that "performs non-parametric estimation of background noise spectrum". It does this by using minimum statistics of the smoothed short-tine Fourier transform (Shafran et al., 2003). They found that the algorithm was effective under different signal to noise ratios. They applied the algorithm on two different videos differing in speaking style and background noise and found that the system was suitable for real-time applications. In our own system the recordings that we will test on were recorded in a controlled environment with limited background noise, so this was not an issue in our own speech detection implementation.

Technological Choices

This chapter will discuss the technological choices for this project and the reasoning behind those choices. The project was implemented solely in Python using JetBrains PyCharm for the IDE. The engagement information detected from the student videos was extracted and written to a csv file for later use when discovering the accuracy of our solution. Jupyter Notebook was using as an online IDE for implementing the machine learning techniques on our dataset.

Python

Python was the choice of language for implementing this project. Many aspects of this project come under the branch of data science including the creation of a large data set, performing machines learning techniques on the data set and analysing the results and accuracy of our system. Python is known as a versatile language that is easier to read when compared to other languages, this comes down to its simpler syntax. With the difficulty that implementing machine learning can bring we decided having clear concise code would be key, it gives the ability to scan code quickly finding any errors. Python is a commonly used language for manipulation of large datasets due to the flexible libraries it offers, libraries used during implementation will be mentioned in the next sub-chapter and more thoroughly discussed during the implementation chapter. The simplicity of Python was an advantage, during implementation any libraries used that were previously unknown to us had a low learning curve and excellent documentation. Python was also used for implementation of the student engagement detection system. The implementation included face detection which using OpenCV can be done in just several lines of code displaying ease of use.

Libraries

As previously mentioned, a vital reason for using Python was because of the libraries offered. This subsection will give a brief overview of the libraries which will be expanded on during the implementation chapter. 7 libraries were used during implementation of the project. The following libraries were used:

OpenCV – a library used for real-time computer vision. The library was originally developed by Intel, but later supported by Willow Garage and then Itseez. During the project it was used for detecting a face (including landmarks such as eyes, nose, mouth) during a video clip.

Mediapipe – an opensource framework for building ML pipelines for processing videos. During this project it was used for detecting poses/gestures this included detecting hands appearing on the screen and keeping a live count of actions.

FER – a is a library that was used for facial expression recognition during this project. The library was authored by Justin Shenk and is available for free use. The library follows a Keras model.

Gaze_Tracking – a library created by software developer Antoine Lame. The library provides an eye tracking system using the webcam of your device. Using the library, we were able to determine the position of pupils and direction of gaze, important parts of our project.

Speech_Recognition – a library created by Anthony Zhang, available for free use. This library was used for converting speech to text so that the number of words spoken during a video could be counted.

MoviePy – a library created by Zulko, 2017, available for free use. This library was used to select video that would have their audio extracted.

CSV – this library was used so that CSV files could be read and written during the project.

Jupyter Notebook

Jupyter Notebook is a web-based platform that allows us to write code into cells and run the cells of code in union or independently. The platform was used for viewing and analysing our dataset so that the accuracy could be discovered. Large datasets can be easily read, displayed, and modified using this platform. We were able to check data types of our data ensuring no changes needed to be made to align the data before ML techniques were applied.

Project Plan

This chapter will discuss the project plan, this includes the timeline and plan for how the project would be completed on time.

Project Timeline – with Gantt Chart

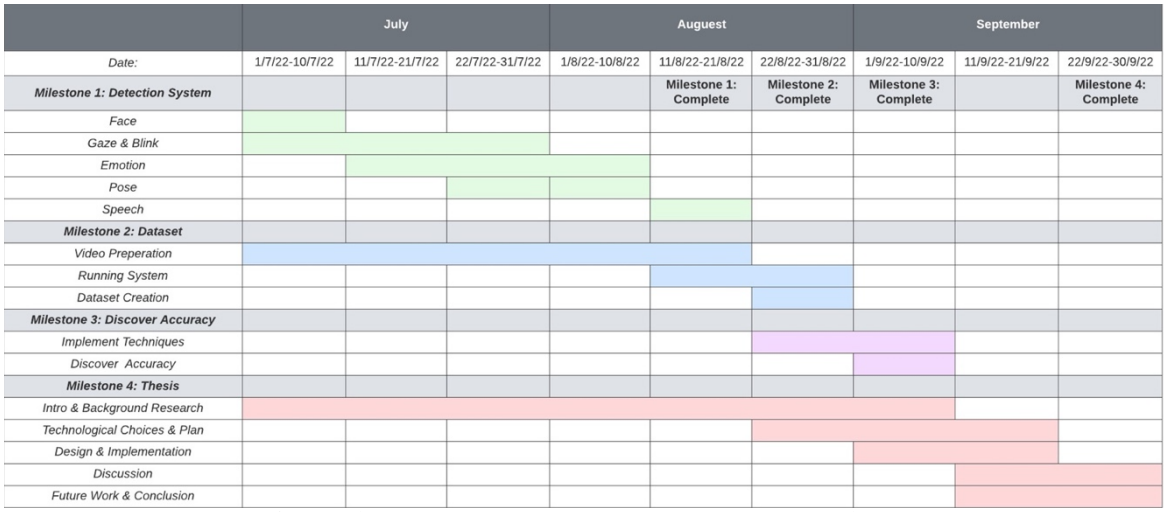


*Figure 2: Gantt chart displaying project timeline *months broken into 10-day weeks**

Project Scope and Background Research

The first step of the project was to understand our aims and area of research. The initial document and specification document helped with this however more research was needed in our specific areas. By viewing figure 2, our project Gantt chart, we can see writing of the introduction and background research started at the beginning of the project. As research was being made from the start and throughout the project this was started early so, we could write specific parts whilst it was fresh in our mind. The background research included reading academic papers on the following topics: robots in education, defining engagement, engagement behaviours and detection, emotion, gaze, blink, pose and speech detection, all important parts of our project. A literature review was performed on multiple papers for each topic with current solutions discussed, the papers results and any limitations they may have that we aimed to improve on.

Milestone 1: Detection System

Creating our engagement detection system meant implementation of the following features: gaze and blink detection, emotion detection, pose detection and lastly, speech detection. These are all discussed in technical detail in the implementation section, but we can view the timeline by using figure 2, all dates are listed in weeks. The first week we focussed on detecting a face, this built foundations for detecting eyes, gaze and poses (1/7/22-10/7/22). Different solutions were tried such as applying face mesh in Mediapipe but we decided to go for Open-cv. Likewise for gaze and blink detection different solutions were tried, this included pygaze and creating our own gaze system. However, the gaze_tracking library created by Antoine Lame had all the functionality we needed for the project and was selected. The gaze and blink features were implemented over the first 3 weeks (1/7/22-31/7/22), this stage was a slower process as we tried to understand the logic behind eye detection – watching YouTube videos to explain this process. Next, the pose feature was implemented from week 3 to 5 (22/7/22-21/9/22). With experience using Mediapipe pose detection was easier to understand and implemented quicker than the previous part. Lastly, speech detection was implemented during the fifth week (21/8/22). The speech detection library used had excellent documentation and was simple to understand so we were able to implement this within a few days. All detected values were written to a csv and with that the system implementation was complete, this brought us to the end of milestone 1.

Milestone 2: Dataset

After receiving the videos to be used by our system from my project mentor the preparation on them began immediately. Our aim was to collect 225 five second clips for to display a student with low, medium, and high engagement (75 for each). This was a length process as it was all done manually, some videos did not fit our purpose, so they were discarded e.g., student not on screen or problems with the tablet/robot. This stage has been listed from week 1 until the end of week 5 as the videos were being discussed in meetings and viewed to see if they would be applicable for this project.

During week 5 and 6 (11/8/22-22/9/22) we ran our system on the videos and collected all the data in csv files. At first these csv files were separate, so an online combiner was used to bring all the data to one file. This process is listed over two weeks as we were first testing the system on just a few videos but expanded to increase our dataset and future accuracies. With completion of this stage milestone 2 was completed by the end of the fifth week (31/9/22).

Milestone 3: Discover Accuracy

To complete the third milestone, we needed to discover the accuracy of our system, this meant implementing different ML techniques and running them on our dataset. From the 22nd of August to the 10th of September, we implemented 7 different techniques for finding accuracy. With previous machine learning experience, this stage ran very smoothly. Any issues encountered were solved by using the documentation provided for each library. Scikit Learn was the library used for ML implementation, a free-to-use library created by David Cournapeau. A brief overview of the steps to find the accuracy were first to import the libraries needed, secondly to split the data into and X and y value, thirdly splitting the values into train and test values, fourthly loading in the ML technique, fitting, and predicting and lastly printing the score with the accuracy score function. The technical process will be discussed further in the implementation chapter. We kept a record of all accuracies and continued to add to our dataset to improve the scores, once we were happy with them the third milestone was complete.

Milestone 4: Thesis

Our fourth and final milestone was to complete a document on our project – the thesis. Writing of our thesis began at the start of our project (1/7/22) with background research being made and discussed during our early chapters as well as making a note of any references we had used. After finishing implementation of the engagement detection system (21/8/22), we began to discuss the technological choices made for our project, this included the main language we used, and any

libraries used. At this point we were also able to discuss the project plan and how we were able to meet our deadline.

During and following the completion of our third milestone we began to write the design and implementation chapter as well as the discussion of our results (1/9/22-21/9/22). Most chapters were touched up on even after we thought we had finished them. The main writing of the thesis was done through the month of September, our Gantt chart showing some indication of the original timeline. The final chapters, future work, and conclusion were written during the final 2 weeks of our project. This was purposely done as we wanted to look at our project, seeing if we had met our aims and objectives. We want to see if there were any limitations or things we could improve on. With completion and submission of the thesis the fourth and final milestone had been reached and this was the end of our project.

System Design and Implementation

This section will be broken down into the system design and the system implementation. The design being a crucial part of understanding the project and its limitations, and the implementation being a thorough technical description of how the project was implemented, along with construction of the dataset.

Design

Here the design of the system will be discussed along with how we got the videos used to create our dataset.

Block Diagram

When designing the system that would be implemented, we created a block diagram. This was to help select components and visualise the system. We can see that the input is a video and that 5 components would be detected and extracted from the video. These being, the emotional state, direction of gaze, number of blinks, number of poses and lastly the number of words spoken. These components were chosen as signs of engagement from extensive research in the area, which has already been discussed in the background research.

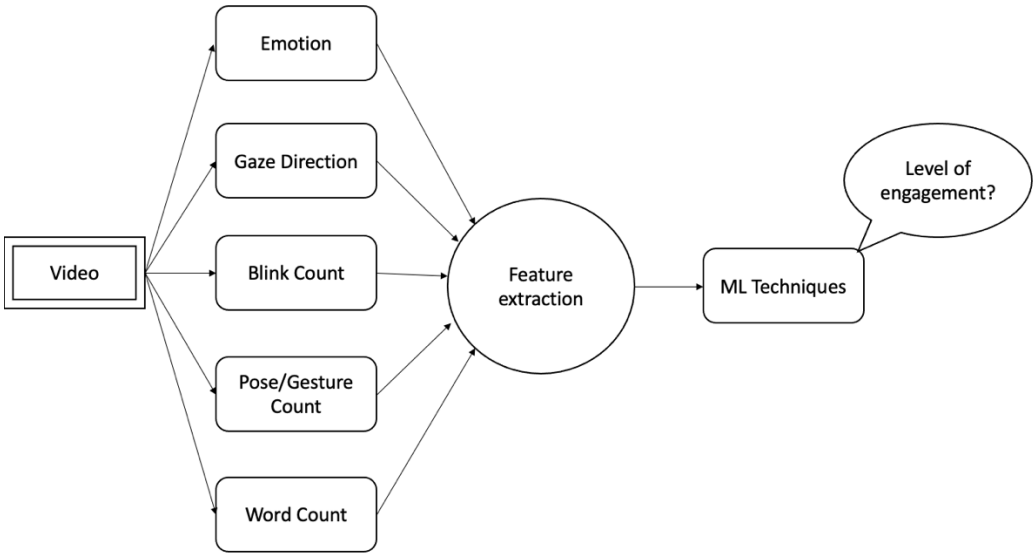


Figure 3: Block diagram of engagement detection system.

Feature extraction takes place using different libraries, further discussed in implementation. The results of the extraction are written to a csv file, creating a dataset. Machine learning techniques are then used on the dataset to find the accuracy of our system when detecting student engagement. The machine learning techniques to be used are the following: Naïve Bayes, Linear Regression, Decision Tree, Random Forest, AdaBoost, Neural Network and KNN. With the results of the ML techniques, we can discover if the level of engagement can be accurately detected.

Dataset

The videos that were used by our system to create our dataset were provided by my mentor Dr Muneeb Ahmad. They were recorded during a study by Dr Ahmad researching social engagement during long-term children-robot interaction (Ahmad et al., 2017). The study was conducted on three groups of students (of children age) who played a snakes and ladders game during a session with a NAO robot (Ahmad et al., 2017). They discuss how the NAO robot would perform game-based Adaptations, emotion-based Adaptations, and memory-based Adaptations, with emotion and then memory being the most effective. This dataset was a good choice for our project as the students in many of the videos were very animated, showing emotions, different poses etc, meaning we could collect rich data on engagement. We also had many different students with differing features, skin tones, hair colours meaning the system would not just have one perspective of engagement.

Implementation

The implementation discusses the full implementation of the project. This will include how the videos used were prepped and selected, the implementation of the engagement detection system, how our dataset was formed and lastly the implementation of different ML techniques for finding the system accuracy. Some examples of system working have been added to the appendix.

Dataset Preparation (Videos)

During the design section we mentioned how we collected the videos that our system would be used on. Here we will discuss how the video were selected and modified to be used by our system to create the dataset.

To start with we had 79 videos of different students interacting with a tutor robot (by playing a game), the videos ranged from just a minute to 10 minutes. Our aim was to collect 5 second clips from the videos with the student showing different levels of engagement: low, medium, or high. To ensure our system detected engagement accurately the shortened clips of video were selected by eye and intuition, deciding which clip fit into which category of engagement. 70 clips were selected for each level of engagement giving us a total of 210 clips.

Student Engagement Detection System

The student engagement system included detecting a face, position of the eyes and the gaze direction, the emotion displayed, a blink counter, detecting and counting poses and lastly the words spoken. During this section we will discuss how all these components were implemented with each main part of implementation having their own subsection.

Gaze Tracking and Blink Count

To implement the gaze tracker, we first had to locate the eyes on a face. Before selecting the gaze_tracking library to do this we investigated different libraries. PyGaze is an open-source toolbox for tracking eyes in Python and was originally considered, however gaze_tracking had more intuitive code and documentation, including individual functions for each direction the eye can gaze, these being; centre, left and right. Our gaze tracking worked by counting each time the student looked left, right, centre, down, or no eyes detected. We then combine certain counts to see which count was the highest and could identify if the student was looking at the robot (straight ahead), at the tablet (down) or not engaged (away from robot and

tablet). To implement this into our project, first variables were created to count the gaze direction and the video to be analysed was loaded in. We had a while loop that would continue to run until there were no frames of the video left. The frame would be read and parsed into the refresh method which would pass the frame ready for analysis. The video would be converted to greyscale for faster processing creating a more optimised system. The gaze_tracking methods were then used to identify which direction the student was looking. If they looked left, the 1 would be added to the left counter, if right, 1 would be added to the right counter and so on. Text was also displayed onscreen so in live time you can follow. In our system if the student was looking down, they would be engaged with the tablet but the gaze_tracking library did not have a down function. Therefore an "is_down()" function was implemented by checking the student is not looking right, left, centre or blinking and that the vertical ratio was not greater than 0.5.

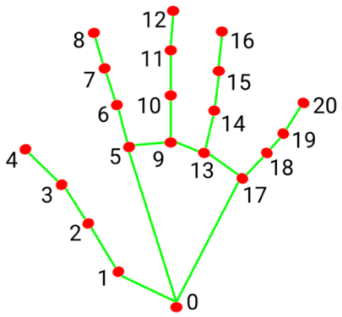
To find the overall gaze for a video our counts were combined and compared in the following way:

- If the students combined right and left count was greater than the blink count combined with the centre count, then the gaze direction was set to 3 (away from tablet and robot).
- If the students combined down count and blink count was greater than the right count combined with the left count, then the gaze direction was set to 2 (looking at the tablet).
- If the students combined blink count and centre count was greater than the right count combined with the left count, then the gaze direction was set to 1 (looking at the robot).

When first implementing the blink count an issue arose, when the eye closed/blinked the count increase indefinitely. We discovered this during early testing of the system when finding the blink count registering over 100 blinks in a 5 second video. To fix this a Boolean check was added. If a blink was registered the "stage" variable – to keep track of stage blink was at, was set to "blinked". When the eye opened and "not blinking" was detected the stage variable was set to "done" and the blink count was incremented by 1.

Detecting and Counting Poses

To detect if a student was posing or not, we first had to detect their hands. To do this we used the Mediapipe library which provides a real-time hand and finger tracking solution. It works by employing machine learning to "infer 21 3D landmarks of a hand from a single frame" (Mediapipe, 2022). The pipeline consists of multiple models working together including a palm and hand detecting model. Hand landmarks can be seen in the figure below, 30,000 real world images have been manually annotated with these landmarks to ensure truth in the model.



0. WRIST
1. THUMB_CMC
2. THUMB_MCP
3. THUMB_IP
4. THUMB_TIP
5. INDEX_FINGER_MCP
6. INDEX_FINGER_PIP
7. INDEX_FINGER_DIP
8. INDEX_FINGER_TIP
9. MIDDLE_FINGER_MCP
10. MIDDLE_FINGER_PIP
11. MIDDLE_FINGER_DIP
12. MIDDLE_FINGER_TIP
13. RING_FINGER_MCP
14. RING_FINGER_PIP
15. RING_FINGER_DIP
16. RING_FINGER_TIP
17. PINKY_MCP
18. PINKY_PIP
19. PINKY_DIP
20. PINKY_TIP

Figure 4: 21 hand landmarks – Mediapipe.

Using this in our implementation we again looped until there were no frames left of the video. Functions "min_detection_confidence" and "main_tracking_confidence" were set to 0.5 and were used to ensure hands were accurately being identified. A frame would be read, converted from BGR to RGB and processed to find hands in the frame. Images were marked as non-writeable before processing to improve performance, then made writeable once processing had finished. Annotations were draw onto the detected hands so we could follow along in real-time. In our system a pose was counted if a student's hands appeared on the screen. If the student's hands stayed onscreen for multiple seconds, this would still be counted as on pose. A Boolean check was implemented to ensure poses were counted correctly. If a student's hand came onscreen then offscreen and again onscreen, this would count as two poses. The pose count is displayed onscreen along with the annotated landmarks for real-time viewing.

Audio Detection and Word Count

The next part of the project was to implement the audio detection and word counter, this was an important part as the student speaking shows engagement. First, the audio was extracted from the m4v video and saved to a wav file. To find the words spoken in the videos wav file first a variable was set to count the number of words spoken. Next, a recognizer was loaded in and assigned to a variable, audio text was identified using a listen method from the speech_detection library. The audio text was then converted into text to read by using the recognize_google method which identified words by checking they appear in the libraries database. The text was split so each word was independent then we were able to count the number of words. Some videos had no audio, so the text detection method was put inside a try and except case to handle errors (videos with no sound).

Emotion Detection

To find the emotion shown during a video the FER library was used. FER splits videos up into frames and uses OpenCV's Haar Cascade classifier to identify faces. Faces can show 6 emotions, these being angry, disgust, fear, happy, neutral, sad and surprise. The library uses a Keras model built with convolutional neural networks. During a video a score will be collected for each emotion, the final emotion is decided by which is highest. To implement this into our system variables were created for each emotion. The emotions were given a number as shown in the table below:

Angry	Disgust	Fear	Happy	Neutral	Sad	Surprise
1	2	3	4	5	6	7

Table 1: Emotion Key

Then after a face had been detected in the frame, FER's analyse function was used to identify emotions for each frame of the video. Using the panda's library, the scores were put into a data frame, and the sum function was used to total the scores. If and "elif" statements were used to compare the different scores to find the emotion with the highest score for the video.

Writing the CSV

The final step was to write all the detected information into a csv so that we could build our data set. An array of titles was created for the header of our csv and the final variables showing scores for each stage of our system were put into an array. Using the csv library, we simply wrote the header and data array into a csv which was saved to our project location. In the figure below we can see an example of low engagement videos being analysed by our system.

engagement

Video_Engagement	Emotion	Gaze	Blink_Count	Pose_Count	Word_Count
1	4	2	6	3	0
1	1	2	0	10	3
1	5	1	0	4	0
1	5	2	5	2	0
1	5	2	2	5	0
1	5	2	4	2	4
1	5	2	6	4	2
1	4	1	16	0	0
1	0	2	2	3	5

Figure 5: Output of student engagement detection system

Dataset and Accuracy

The dataset was created following the completion of the student engagement system and the video preparation. We ran the system on 90 videos which had been self-described as one of the following, low engagement, medium engagement, or high engagement (225 in total). Following this process, we had a vast dataset full of values representing different types of engagement. Originally the dataset contained some missing values for emotion, this was due to the student's head facing away from the camera for parts of the clip. To fix this we manually went through videos where this was the case and gave the student an emotion score based off what we saw.

With the completed dataset we were able to begin the last stage of our project by finding the accuracy of our system. To do this we employed the following

machine learning algorithms: naïve bayes, linear regression, decision tree, random forest, Ada boosting, neural network, and k-nearest neighbour. We decided to use a range of machine learning techniques to be sure that our model was somewhat accurate. If we had just used one technique and gotten a high accuracy this would not be an indicator of an accurate system.

Libraries (ML and Accuracy)

Classification Report – Used to output a classification report for each ML technique.
Train Test Spilt – Used to split data for training and testing**.**
Gaussian NB – Provided us with a naïve bayes classifier.
Linear Regression – Allowed us to create a LR model.
Random Forest – Provided us with a random forest classifier.
Tree – Provided us with a decision tree classifier.
Ada Boost Classifier – Provided us with an Ada boost classifier.
Neural Network MLP Classifier – Provided us with a neural network classifier.
K Neighbours Classifier – Provided us with a K-nearest neighbour classifier.

ML Technique Choice Explanation

Here we will discuss why we have chosen the machine learning techniques that we have and their pros and cons. All our methods are supervised learning which is defined "by its use of labelled datasets to train algorithms that to classify data or predict outcomes accurately" (IBM et al., 2020). In our case predicting a level of engagement from a video.

Naïve Bayes is an algorithm based on "applying the Bayes' theorem with the "naïve" assumption of conditional independence between every pair of features give the value of the class variable" (Sagar et al., 2019). The algorithm is known for being quicky and simple with the ability to solve classification problems (VAdapalli et al., 2020). This was nothing but true with the Sklean library allowing us to find an accuracy in just 3 lines of code. The algorithm can also outperform other models with less training data, which in our case was useful as the dataset wasn't thousands

of entries (VAdapalli et al., 2020). A con of the algorithm is that it assumes all features are independent, which usually isn't the case. In our example gaze and blink being dependent on each other. Overall, we found the algorithm to be a good start in finding the accuracy of our system.

The second technique used was linear regression – which performs a regression task, the only technique to do this that we chose. The model is used to find a prediction value based on independent variables. Like our previous technique the model is simple to implement and allows for easy-to-understand outputs, such as the R2 score, variance score, and squared errors. A con of this technique was that anomalies had a large effect on the regression which can cause inaccurate outputs (Rout et al., 2020). This came true in our own predictions only managing an R2 of 0.33. Although the results weren't as high as we had hoped we felt a regressive technique added depth to our search of accuracy.

Decision tree is an algorithm where data is transformed into a tree representation. Internal nodes of the tree represent an attribute and each leaf node a class label. The algorithm does not require normalisation or scaling of data which made it simpler on our end. Another advantage of the algorithm is that missing values will not affect the process of building the decision tree (Dhiraj et al., 2019). This helped finding accuracy on our early dataset that featured missing values, though these were manually fixed at a later point. A disadvantage of the algorithm is that small changes in data can make a big change in the decision tree (Dhiraj et al., 2019). However, as our dataset was not changed frequently, bedsides from more entries being added, our scores remained stable.

Random forest is an algorithm which combines outputs of multiple decision trees to make a single result. It can be used for classification and regression problems but was used for classification in our project. It has become a more popular technique recently with its ease of use and flexibility (IBM et al., 2020). An advantage of using random forest in our case was its ability to be robust to anomalies. Some entries in our dataset showed unexpected values and this algorithm gave us assurance we could still get a good accuracy score. The algorithm can sometimes be slow compared to

others when training but with the size of our dataset this was not an issue for us (IBM et al., 2020).

Ada boost can be used as an ensemble with other techniques, but we used it as an independent classifier. It's defined as a meta-estimator that first fits a classifier on the original dataset and then fits additional copies of the classifier on the same dataset but where weights of incorrectly classified instances are adjusted in a way that following classifiers focus on the more difficult cases (Sklearn et al., 2022). An advantage of the algorithm is outputting more accurate results compared to other models – due to its focus on outliers. However, the algorithm can have issues with overfitting, it needs quality, high volume data which although our data is quality it is not as high volume as we would like. Overall, we chose to use it as we thought it could provide some more accurate analysis of our dataset.

Neural network is an efficient algorithm that can complete tasks quickly (Ayan et al., 2022). The algorithm works by processing data in a way inspired by the human brain, it features nodes in a structure that resemble a brain. They boast continuous learning, it is designed to learn and produce more accurate results as this process goes on. This was a key feature for us as we added more and more entries to our dataset hoping to improve accuracy. They are said to be complex to program but with the use of the sklearn library this was not an issue for us. The algorithms continuous learning is a benefit however its data dependency limits its results with smaller datasets. We decided that the size of our dataset was large enough to output interesting results.

K-nearest neighbour (KNN) is a method for estimating the chance that a data point will become a member of a specific group based on what group the data points nearest to it belongs to (Joby et al., 2021). The algorithm is said to "constantly evolve", immediately Adapting as new training data is collected. It allows quick changes to changes in input in real-time (Genesis et al., 2018). Unlike a model such as linear regression, KNN makes no assumptions which the data must meet before implementation. With its pros also come some cons, the algorithm cannot deal with missing data. The algorithm is also sensitive to outliers due to its picking neighbours

on a distance base. We coded our system not to output null values or missing data with try and catch cases around parts of our code, this meant the missing data problem did not affect our implementation of the algorithm.

Results and Discussion

With completion of the machine learning techniques, we were able to find the accuracy of our system in a range of ways. We will now discuss the results, what they mean and see if we have achieved the aims that we set out at the start of the project.

Classification Report Explanation

For all techniques besides from linear regression we have a classification report. The report will include the precision, recall, f1-score, weighted average, and accuracy.

The precision calculates the accuracy of positive predictions with:

$$P = TP/(TP+FP)$$

The recall calculates the percentage of positive cases caught with:

$$R = TP/(TP+FN)$$

The f1 score calculates the percent of positive predictions that were correct with:

$$F1 = 2*(Recall*Precision) / (Recall + Precision)$$

(Kohli et al., 2019) for calculations.

* TP = Case was positive and predicted positive
* FN = Case was positive but predicted negative
* FP = Case was negative but predicted positive

Each of these will have a weighted average and then one final value for the accuracy.

For linear regression we have provided a mean squared error, variance score, and an R2 score. MSE is the average of the square of errors – error meaning the difference between observed values and predicted ones (Walker et al., 2018). Variance is the measure of how observed values differ from the average of predicted values. The R2 score is defined as "the proportion of the variance in the dependent variable that is predictable from the independent variables" (Walker et al., 2018).

Summary of Results

Technique	Weighted Averages			
	Precision	Recall	F1-Score	Accuracy
Naïve Bayes	0.80	0.80	0.80	0.80
Random Forest	0.80	0.80	0.80	0.80
Decision Tree	0.83	0.82	0.82	0.82
Ada Boost	0.90	0.89	0.89	0.89
Neural Network	0.78	0.78	0.76	0.78
K-Nearest Neighbour	0.80	0.78	0.78	0.78

Table 2: Accuracy of System from ML Techniques

Technique	Mean Squared Error	Explain Variance Score	R2 Score
Linear Regression	0.24	0.6	0.58

Table 3: Accuracy of System from Linear Regression

Above we see a table of our results from running the ML techniques on our dataset. Our highest scores came from the precision, the accuracy of our positive predictions. With ADA boost being our highest at an impressive 0.90 and neural network being the lowest with 0.78. Our other highest scores being decision tree (0.83), naïve bayes, random forest, and k-nearest neighbour all showing 0.80 respectively.

Our recall scores, percentage of positive cases caught, dropped a little in accuracy but still displaying strong numbers with our highest score being 0.89 for ada boost and lowest being 0.78 for neural network and k-nearest neighbour. Our next highest being decision tree with 0.82. Then naïve bayes boost and random forest both returning a score of 0.80, respectively.

Our f1 scores, the percentage of positive predictions that were correct gave further insight into our system accuracy with our highest score being 0.89 with ADA boost and our lowest being neural network 0.76. Decision tree being the second highest with 0.82, naïve bayes and random forest tied again with 0.80 and k-nearest neighbour 0.78.

Finally, we have our accuracy score for each technique, ADA boost coming out with our highest score of 0.89. Decision tree second with 0.82. Naïve bayes and random forest both with 0.80 and lastly, neural network and k nearest neighbour with 0.78. For each score we see neural network and k-nearest neighbour continually come out last. As mentioned in our technique explanation, neural network can struggle with smaller datasets and may have impacted our results here. K-nearest neighbour can struggle with outliers, which we experienced with some of our feature detection, this may have dropped our accuracy score here. Ada boost and decision tree all performed well, coming out with the highest scores continually. Naïve bayes and random forests accuracy performed well too throughout the process. During early analysis of our dataset, it still contained 0s (missing values for emotion) so as we filled in these gaps the accuracy only grew stronger, especially for k-nearest neighbour.

For linear regression we achieved an R2 score of 0.58, a variance of 0.6 and MSE 0.24. An r2 score of greater than 0.7 is seen as showing a high level of correlation so we are a little below that bracket, but over the low bracket of 0.35. We achieved a respectable variance score of 0.6, a variance score of 0.35 is said to show that the data would not be useful to us (Sufiyan et al., 2014). We achieved an impressive MSE score of just 0.24, indicating a good accuracy of 76% from our system (for the size of the dataset).

Discussion of Results

On average our system had an accuracy of 81% and a precision of 82%. Due to time constraints our dataset was limited to 75 videos for each level of

engagement. Our system shows positive results for detecting engagement from a student. Although we had a large dataset (in scale with our project timeline) some ML studies work with datasets containing thousands of entries and more. We believe by increasing our dataset we could continue to improve the accuracy of our detection system. Another way we could have improved our results was by handling outliers. With future use of our system if outliers are encountered, we will manually watch the videos and make sure the detection is correct. Unfortunately, our linear regression scores were lower in comparison to other techniques. This biggest factor of this was the dataset size and perhaps only have 5 variables to work off. To improve this in the future we would like to increase the dataset size as mentioned but also add other variables to be detected by our system, this will be discussed more in future work.

Engagement Behaviours

During our study we selected 75 videos displaying 3 levels of engagement: low, medium, and high. Our system detected gaze, emotion, blinks, poses, and speech and recorded this data in csv files for us to analyse. We will now discuss what we have learnt about behaviours/attributes of engagement and what indicates different levels.

The first thing we found was that emotion seemed to be an important factor in recognising engagement, over 50% of students in the "low" engagement category registered an emotion score of 6 (sad). Whereas, for medium and high engagement over 90% of children registered either a 4 or 5 emotion score indicating happiness or neutral. Although neutral emotion may not normally suggest engagement, we decided it was actually a high factor in students displaying high engagement as a focussed face would not necessarily show a happy emotion.

During gaze detection we registered 85% of students looking away from the tablet or robot, however 15% of students were looking central indicating that although a student is looking at their task they may not actually be engaged. 100% of students with medium to high engagement had a gaze score of 1 (looking at robot) or 2 (looking at tablet). This feature was a high indicator of an engaged student.

During the blink detecting we registered some interesting results. Normally, students with a medium to high engagement score showed a higher count of blinks than low engagement students. However, around 30% of highly engaged students registered 0 blinks, we think this was due to the focus of students and the length of the videos we were detecting from. Only being 5 seconds in length some students would stare intensely at the robot or tablet – indicating high engagement but not blinking in the duration of the video. Interestingly, low engaged students usually registered 1 or 2 blinks during a clip but around 40% registered 0 blinks indicating that although they were looking forward, they may have been zoned out. Another reason for this was that some students with low engagement were not looking toward the robot which is where the camera was located for our experiment. Therefore, there were no blinks to be detected as the student would be looking away with the eyes not insight.

The pose count was quite an ambiguous variable in the sense that both low and high engagement could record a high count of poses. This was because some of the low engagement videos included students looking away and getting up from their seats, by raising their arms multiple times during this it would count as multiple poses. However, many low engagement videos showed students making 0 poses also. Medium to high engaged videos mostly showed at least 1 pose with a high number of poses also common. Some showed 0 poses for the same reason as a low blink count, the student being fully engaged and not moving at all.

Our last behaviour detected was the number of words spoken. We found that over 70% of low engagement videos had 0 words detected and the 30% either being 1 or 2 words. Medium engagement showed a similar pattern, but the word count increased for some entries by 1. Word count showed a high correlation with the highly engaged videos. 100% of videos detected at least one word spoken by the student. However, over 90% of videos showed that the student spoke between 3 and 9 words, indicating that they were highly engaged and conversing with the robot or asking questions about their task.

Permutation importance with random forest was performed on the dataset and showed that word count and the gaze were the highest indicators of high engagement with a weight of 0.73 +- 0.32 and 0.43 +- 0.24, respectively. Next emotion and blink count showing fair levels of correlation but pose performing the worst with a negative correlation of -0.025 +- 0.05.

Closing Discussion

At the beginning of this thesis, we presented our aims as the following:

1. What level of engagement does a student show when interacting with a tutor bot?
2. What attributes suggest a student is engaged and can we detect it accurately?
3. Can we help define student engagement?

Using our system, we have discovered that students can actively engage with a tutor bot and that different levels of engagement can be shown by the student. We have identified what attributes shown by the student indicate that they are engaged at different levels. We discovered that word count and gaze seem to show the highest correlation to a student being highly engaged. We also discovered that pose and blink count don't show a strong correlation with highly engaged students as a similar amount was detected for lowly engaged students.

Using our system, we were able to detect level of student engagement with an average of 81% using 6 different machine learning techniques, some individual techniques producing accuracies of up to 89%. We have met our aim of being able to accurately detect engagement from a student. We believe with further additions to our dataset we would be able to produce an average accuracy of 90%+ over all the machine learning techniques used.

Our third aim was to further help define student engagement. We believe we have done this by furthering the research on what behaviours indicate engagement. We know that words spoken by the student and their gaze are strong indicators of being engaged. (Kelders et al., 2020) review on components of engagement found

cognition and behaviours to be strong indicators of highly engaged students. We have further proved this point with our study and helped clarify what specific behaviours indicate this.

During the project some difficulties occurred with our system. More specifically when detecting student behaviours sometimes due to the camera location we would not be able to constantly detect the student over the full length of the video. This only occurred during some low engagement videos and to counter the difficulty we manually went through impacted videos making note of behaviours to ensure of detection was accurate. In the future to combat this we would like to use more than one camera location during the study, this has been discussed further in future work.

Future Work

During this project we have been successfully detected engagement from a student and what behaviours suggest high/medium/low engagement. With this information we would like to research optimisation of learning with a HCI component. We aim to use tutor robots to optimise student learning by performing lessons and applying reinforcement learning strategies when students react positively to specific methods of teaching. Every student learns in a different way, but this is categorised into 4 main groups (S Team et al., 2020):

1. Visual – students that retain information better when it is presented to them in a graphical way e.g., diagrams and chart.
2. Auditory – students that retain information better when it is presented to them vocally.
3. Reading and Writing – students that retain information better when it is presented to them in the form worksheets and text-heavy resources.
4. Kinaesthetic – students that retain information better when physically taking a role and engaging their senses during work e.g., lab experiments.

Using these 4 groups of engagement we would create four lessons with a robot to discover what type of learner a student is. We could then optimise each student's study session with the aim of keeping them as highly engaged as beneficially possible. Examples of this would be, if we discover a student prefers to learn from auditory stimuli then their lesson would be optimised in this way. They would spend a lesson listening to the tutor bot vocally announce information. If we learn that a student prefers a kinaesthetic approach, we could design a lesson where the student must complete a simple science experiment whilst interacting with the robot, taking instructions, and giving feedback to the robot about their experiments results. During this more optimised approach to learning we would continue to detect the students to ensure high engagement with their tasks.

In relation with the work completed in this project we would like to expand our dataset by performing more HCI experiences for students in schools. In this project

all videos collected were of students playing snakes and ladders with a robot. In the future we would like to create more games/lessons for the students to interact with. By continuing to add to our dataset we aim to increase the accuracy of our detection system. We would also like to implement more accurate detection by using more cameras. In this project we had one camera that would record the child from the robot's perspective. By adding more cameras to our study, we could follow the students face throughout the whole interaction which would improve our accuracy score further.

Conclusion

With this project we aimed to further understand human-computer interaction but more specifically, a student interacting with a tutor bot. With our project we had a global aim of eventually helping the 69 million demands for teachers and tutors. Although this would take many more years of research and development, we believe that our core aims have been reached.

On the technical side we aimed to implement a software solution that could detect a student's gaze direction, number of blinks, emotion, number of poses, and number of words spoken during a video. We successfully implemented this system and were able to use it on 225 videos of students interacting with a robot. Following this we were able to categorise the videos into either low, medium, or high engagement. With this we also met our second technical aim of creating a plentiful dataset. Using the data, we were able to reach our third aim of performing 7 machine learning techniques on the dataset to discover the accuracy of our system.

Our non-technical aims were to first find the engagement levels of a student when interacting with a robot. Secondly, to find out what behaviours indicate different levels of engagement and if we can detect it accurately. Lastly, we aimed to help further define student engagement. During our discussion we thoroughly discussed how we believe that we achieved these aims with great success. Using our system, we were able to detect three levels of engagement with an accuracy of 81% over 6 classification machine learning techniques. With ADA boost we were able to achieve a precision high of 90% and an overall accuracy of 89% with the technique. We have noted what behaviours by the students suggested high levels of engagement and further backed this up by performing permutation importance with random forest to discover word count and gaze as the highest indicators.

During future work we have discussed the limitations of our project and where we would like to take it with further time and research. Specifically, into the field of learning optimisation by using reinforcement techniques after discovering what type of learner a student is.

Finally, we have further helped define student engagement. We have discovered that along with cognitive components, behavioural components such as the ones we have detected can provide a strong indication of a student's engagement whilst interacting with a robot. With our research we have done so with a high level of accuracy which has been displayed in this thesis.

Bibliography

Degree Choices Staff, *Which teachers are most in demand?* 2021. Accessed on 22/02/22. [Online]. Available: https://www.degreechoices.com/blog/teachers-in-demand/

L. Merkle, *Measuring the effectiveness of robots in teaching computer science*. Accessed on 22/02/22. [Online]. Available: https://www.academia.edu/21974347/Measuring_the_effectiveness_of_robots_in_teaching_computer _science

Saskia M. Kelders, Llewellyn Ellardus van Zyl, and Geke D. S. Ludden, *The Concept and Components of Engagement in Different Domains Applied to eHealth: A Systematic Scoping Review*. Accessed on 5/06/22. [Online]. Available: https://www.frontiersin.org/articles/10.3389/fpsyg.2020.00926/full

Katharina Schnitzler, Doris Holzberger, and Tina Seidel, *All better than being disengaged: Student engagement patterns and their relations to academic self-concept and achievement*. Accessed on 05/06/22. [Online]. Available: https://link.springer.com/article/10.1007/s10212-020-00500-6

P. Sharma, S. Joshi, S. Gautam S. Maharjan, V. Filipe, and M. Cabral Reis, *Student Engagement Detection Using Emotion Analysis, Eye Tracking and Head Movement with Machine Learning*. Accessed on 26/02/22. [Online]. Available: https://arxiv.org/abs/1909.12913.\

M. Ali Akber Dewan, M. Murshed and F.Lin, *Engagement Detection in Online Learning: a review*. 2019. Accessed on 26/02/22. [Online]. Available: https://link.springer.com/content/pdf/10.1186/s40561-018-0080-z.pdf

O. Mubin, C. Stevens, S. Shahid, A. Al Mahmud, and J. Dong, *A Review Of The Applicability Of Robots In Education*. 2013. Accessed on 26/02/22. [Online]. Available: https://d1wqtxts1xzle7.cloudfront.net/45288492/A_review_of_the_applicability_of_robots

L. Toh, A. Causo, P. Tzuo, I. Chen, and S. Yeo, *A Review of the Use of Robtos in Education and Young Children*. 2014. Accessed on 26/02/22. [Online]. Available: https://www.jstor.org/stable/jeductechsoci.19.2.148

T. Belpaeme, J. Kennedy, A. Ramachandran, B. Scassellati, and F. Tanaka, *Social robots for education: A review*. 2018. Accessed on 26/02/22. [Online]. Available: https://www.science.org/doi/10.1126/scirobotics.aat5954

E. Konijn, M. Smakman, and. R. Berghe, *Use of Robots in Education*. 2020. Accessed on 08/08/22. [Online]. Available: https://www.researchgate.net/profile/Elly-Konijn/publication/345737267_Use_of_Robots_in_Education

MP, *Media Pipe hands. 2022*. Accessed on 08/08/22. [Online]. Available: https://google.github.io/mediapipe/solutions/hands.html

S. Jaiswal and G. C. Nandi, *Robust real-time emotion detection system using CNN architecture*. 2019. Accessed on 6/07/22. [Online]. Available: https://link.springer.com/article/10.1007/s00521-019-04564-4

G. Rogez, J. Rihan, S. Ramalingam, C. Orrite, and P. Torr, *Randomised trees for human pose detection*. 2008. Accessed on 4/08/22. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/4587617

S. Obdrzalek, G. Kurillom J. Han, T. Abresch, and R. Bajcsy, *Real-Time Human Pose Detection and Tracking for Tele-Rehabilitation in VR*. 2012. Accessed on 05/08/22. [Online]. Available: Real-time human pose detection and tracking for tele-rehabilitation in virtual reality

Shafran and R. Rose, *Robust speech detection and segmentation for real-time ASR applications*. 2003. Accessed on 08/08/22. [Online]. Available at: 10.1109/ICASSP.2003.1198810

M. Ahmad, O. Mubin, J. Orlando, *Daptive Social Robot for Sustaining Social Engagement during Long-Term Children-Robot Interaction*. 2017. Accessed on 10/09/22. [Online]. Available at: https://www.researchgate.net/publication/316354658_Daptive_Social_Robot_for_Sustaining_Social_Engagement_during_Long-Term_Children-Robot_Interaction

IBM Cloud Education, *Supervised Learning*. 2020. Accessed on 15/09/22. [Online]. Available at: https://www.ibm.com/cloud/learn/supervised-learning

R. Sagar, *What is a naïve bayes classifier and what significance does it have in ML*. 2019. Accessed on 25/09/22. [Online]. Available at: https://analyticsindiamag.com/what-is-a-naive-bayes-classifier-and-what-significance-does-it-have-for-ml/#:~:text=Naive%20Bayes%20methods%20are%20a,value%20of%20the%20class%20variable.

S. Kohli, *Understanding a Classification Report for Your ML Model*. 2020. Accessed on 25/09/22. [Online]. Available at: https://medium.com/@kohlishivam5522/understanding-a-classification-report-for-your-machine-learning-model-88815e2ce397

P. Vdapalli, *Naïve Bayes*. 2020. Accessed on 25/09/22. [Online]. Available at: https://www.upgrad.com/blog/naive-bayes-classifier/

K. Dhiraj, *Pros and Cons of Decision Tree Algorithm*. 2019. Accessed on 25/09/22. [Online]. Available at: https://dhirajkumarblog.medium.com/top-5-advantages-and-disadvantages-of-decision-tree-algorithm-428ebd199d9a

IBM Cloud Education, *Random Forest*. 2020. Accessed on 25/09/22. [Online]. Available at: https://www.ibm.com/cloud/learn/random-forest

Sklearn, *AdaBoostClassifier*. 2022. Accessed on 26/09/22. [Online]. Available at: https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.AdaBoostClassifier.html

myAyan, *Neural Networks Advantages and Disadvantages*. 2022. Accessed on 26/09/22. [Online]. Available at: https://www.myayan.com/advantages-and-disadvantages-of-neural-networks

Sufiyan, *Low Squared R Values, Regression*. 2014. Accessed on 27/09/22. [Online]. Available at: https://www.researchgate.net/post/Low-R-squared-values-in-multiple-regression-analysis

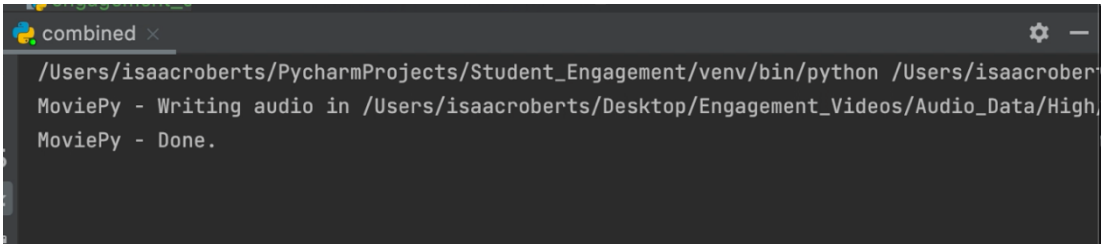
S. Team, *4 Types of Learning Styles*. 2020. Accessed on 28/09/22. [Online]. Available at: https://sphero.com/blogs/news/learning-styles-for-kids

Appendix

Extra points:

- During the study some videos were filmed at different angles which affected our gaze detection. To fix this problem the videos were manually watched to confirm gaze direction. The dataset was then updated with the correct value if was needed.

Pictures of detection System



Item 1: Audio being extracted for speech to text word detection

Item 2: Gaze and blink being detected

Item 3 and 4: A pose being detected and the count increasing