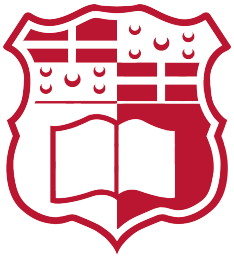


An Intelligent Virtual Teaching Environment with Emotional Recognition: Supporting Novice Teachers through LLM-Driven Multi-Agent Simulations

JOHN CARMEL AQUILINA

Supervised by Dr. Vanessa Camilleri

Faculty of Information & Communication Technology

University of Malta

January, 2026

A dissertation submitted in partial fulfilment of the requirements for the degree of Master of Science in Artificial Intelligence.

Dedicated to my family, Nicholas Aquilina, Elizabeth Aquilina, Lisa Marie Aquilina, Michael Caruana, and to my boyfriend, Ben Nwoko, for supporting me throughout my work.

Acknowledgements

I cannot express enough how grateful I am to my tutor, Dr. Vanessa Camilleri, for the guidance and help she has given me throughout this dissertation. Additionally, I extend my gratitude to Prof. Matthew Montebello for his support.

I am also grateful to the teachers and educational experts who participated and helped in my research. Without their feedback, the implementation would not be where it is now. I'm also very thankful for the teachers' taking the time out of their schedule to meet up and participate in the experiments.

Lastly, I am also very grateful for my family, my boyfriend, and my friends who always believed in me and supported me. Their moral support has managed to keep me going when times got tough, and I will forever be grateful for their never-ending love and support.

Abstract

Novice teachers often report insufficient training in managing disruptive classroom behaviours, significantly undermining learning environments. This dissertation addresses this gap by developing Class Simulation, a Proof of Concept (POC) virtual teaching framework that employs Large Language Models (LLMs) to create interactive training scenarios. Implemented in Unity, the system integrates three components: an Orchestrator that generates behaviourally realistic classroom situations, a Student Agent that simulates learners with emotional states, and an Analyser that provides feedback and clarification on teacher actions.

Given the absence of LLMs tailored for educational contexts, we designed a fine-tuning pipeline that transforms data from educational professionals into domain-specific training material for the Orchestrator and Analyser. To evaluate the system, educational experts and similarity metrics were used to assess the accuracy and pedagogical validity of generated scenarios, student emotions, and feedback, while teachers participated in a usability study to gauge practical value.

Findings indicate that: experts equally preferred both scenarios generated from the Orchestrator and non-fine-tuned models, whilst metrics slightly favoured the Orchestrator (Average Bilingual Evaluation Understudy (BLEU) score of 0.1835 vs 0.1312 in favour of the Orchestrator), experts and metrics both predicted the correct emotions used in the Student Agent two out of three times, and experts and metrics slightly preferred the feedback generated by the Analyser (average BLEU score of 0.0127 vs 0.0098). Teachers valued the system as a safe environment to practise behaviour management strategies and emphasised its potential to complement traditional training. Key limitations include the small dataset, scalability challenges, and reliance on commercial LLM APIs.

Despite these constraints, the results demonstrate the feasibility of combining LLMs, emotion modelling, and multi-agent simulation to support teacher preparation. This work lays the foundation for future research on scalable, domain-adapted AI systems in education.

Contents

List of Abbreviations

List of Figures

List of Tables

1

Introduction

Large Language Models, a groundbreaking development in Artificial Intelligence (AI), have significantly altered our approach to understanding and generating natural language. Central to these advancements is the Transformer Architecture (Vaswani et al. (2017)), which has reshaped applications in Natural Language Processing (NLP) such as language modelling, machine translation, and text generation. Beyond traditional NLP tasks, researchers are exploring the use of LLMs in various fields, such as education; automatically detecting and correcting grammatical errors in text, reading assistance through text simplification and read-ability assessment, and Intelligent Tutoring Systems (ITSs) which provide learners with one-on-one tutoring. Their adoption by Education Technology (EdTech) companies such as Duolingo (Naismith et al. (2023)) and Grammarly (Raheja et al. (2023)) shows the strong impact the recent advances in LLMs are having on the development of educational applications (Alhafni et al. (2024)). This dissertation investigates the application of LLMs in three different cases: 1) the generation of teaching scenarios to portray in-class disruptive behaviour, 2) the generation of realistic student actions and responses, and 3) the generation of analysis on teacher actions to provide feedback to the teacher.

Given the prevalence of children with disruptive behaviours in the classroom, it is crucial that teachers are adequately trained to manage these situations effectively (Stevenson et al.). Without proper training, such behaviours can disrupt the learning environment, affecting both the students involved and their peers (McGuire et al.). The use of evidence-based training programs for teachers to manage disruptive behaviours in students can help to improve their preparedness when handling similar scenarios (Gettinger et al. (2008)). This research builds upon these programs by using LLMs to simulate classroom scenarios involving students with disruptive behaviours. These simulations allow teachers to practice and refine their responses to disruptive behaviours in a controlled environment.

1.1 | Problem Definition

This dissertation addresses two problems concerning the use of LLMs to simulate classroom scenarios with disruptive behaviours. First, we address the novice teachers' unpreparedness in managing behavioural challenges in real classrooms (Cooper et al. (2018); Mireles-Rios et al. (2019)). New teachers often express a need for clearer expectations and more opportunities for guided, hands-on experience (Bruner (2009); Piaget (2013); VYGOTSKY (1978)). Shank and Santiague (2022) points out a lack of practical, evidence-based training and limited preparation for addressing disruptive behaviours.

Second, we address the lack of AI implementations in simulating classroom scenarios with disruptive behaviours. Whilst LLMs are powerful for general applications, they require significant adaptation to generate domain-specific educational scenarios that authentically represent classroom dynamics with disruptive student behaviours. We tackle this by defining a fine-tuning pipeline using authentic classroom management data.

Beyond generating realistic scenarios, our system must overcome LLM's tendency toward character inconsistency when fulfilling the Student Agents role. This necessitates implementing a contextual memory system which stores and retrieves relevant past interactions to maintain behavioural consistency.

A critical challenge of the analysis component includes the need to ensure factual accuracy in the feedback provided to teachers. Within our approach, we implement a data extraction mechanism that verifies whether sufficient contextual information exists before generating analytical feedback. This minimises the risk of fabricated or hallucinated content being generated which could undermine the system's educational value.

Finally, the system requires integration and communication of multiple components within a virtual environment. This includes scenario generation, student behaviour simulation, and teacher action analysis and feedback generation. This challenge involves creating interfaces which allows displaying of the generated scenario, the emotions and responses generated by the student agents, and the feedback generated based on the teachers' actions.

1.2 | Aims and Objectives

This research aims to develop and evaluate a Large Language Model-based simulation system that enhances teacher preparedness for managing disruptive classroom behaviours through realistic scenario generation, student behaviour simulation, and actionable feedback analysis.
Our investigation centres on two primary research questions:

1. *Does the class simulation system provide pedagogically valuable and implementable feedback that impacts teachers' classroom management strategies?*
2. *To what extent does our implementation generate scenarios, student behaviours, and feedback that align with educational professionals' expectations?*

To investigate these research questions, this study will follow these objectives:

1. Select and optimise a single appropriate Large Language Model for each system component (Orchestrator, Student Agent, and Analyser) based on its capability to generate educationally relevant content, maintain contextual consistency, and provide pedagogically sound feedback;
2. Develop a specialised dataset through collaboration with experienced educators, capturing authentic classroom management challenges and effective intervention strategies;
3. Optimise LLM fine-tuning parameters through iterative testing to maximise realism in scenario generation, and pedagogical validity in feedback analysis;
4. Design and implement an integrated virtual environment facilitating natural interaction between teachers and all system components — Orchestrator, Student Agents, and Analyser;
5. Conduct a comprehensive evaluation of the system through multiple assessment strategies, including expert validation by classroom management specialists, user experience metrics from participating teachers, and technical performance analysis to ensure reliability and authenticity across all system components.

1.3 | Overview of the Proposed Solution

To address the issue of teachers being insufficiently prepared to manage disruptive classroom behaviour, we present our POC called **Class Simulation**. It consists of three main components: (1) the **Orchestrator**, which is prompted to randomly generate realistic classroom situations involving students with disruptive behaviour; (2) the **Student Agent**, which imitates the actions and behaviours of various types of students; and (3) the **Analyser**, which evaluates how the teacher handles the generated situations, provides feedback, and offers additional support in the form of a mentor when the initial analysis is not clear enough. For this solution, we select the appropriate LLM to handle the aforementioned tasks, and fine-tune it with real-world data to ensure the Orchestrator and Analyser generate realistic scenarios and actionable feedback.

1.4 | Contribution

This dissertation makes several significant contributions to AI applications in educational technology. At its foundation, we develop a pipeline for fine-tuning LLMs in educational contexts. This process includes gathering data (with the help of teachers), creating the fine-tuning data from the gathered data, and fine-tuning the chosen model. This results in domain-optimised models capable of generating scenarios and feedback similar to real-life.

Building on this foundation, we advance the field through our implementation of a fine-tuned LLM that generates realistic classroom scenarios and seamlessly translates them into interactive virtual environments using Unity. This contribution bridges the gap between NLP and spatial representation of educational settings.

The research further contributes an innovative feedback system through a specialised LLM fine-tuned to analyse teacher responses to disruptive classroom situations. This system not only provides structured feedback on classroom management strategies but also functions as an interactive mentor, allowing teachers to seek clarification through natural language dialogue involving text.

Our work also offers methodological advancements in emotional intelligence for AI agents through the Student Agent's processing pipeline. This system interprets teacher

inputs, performs sentiment analysis with intensity detection, calculates appropriate emotional state transitions, generates contextually relevant student actions, and communicates these through both verbal responses and visual emotional displays.

Finally, we contribute a cohesive virtual environment architecture that integrates these AI components into a holistic teacher training experience. This environment facilitates structured interactions with the Orchestrator for scenario generation, the Student Agent for realistic behavioural simulations, and the Analyser for pedagogical feedback and mentorship, creating a platform for practising and refining classroom management skills.

1.5 | Document Structure

The rest of this dissertation is structured as follows.

Chapter 2. Background: We give a detailed background on LLMs and their advancements throughout the years. We also go into detail on the various prompting methods used. Finally, we detail several implementations of LLMs as agents for various roles.

Chapter 3. Literature Review: We provide information on valuable literature which helped our solution. We go through the implementations of AI in education and educational simulations. Furthermore, we go through LLMs in education, shifting our focus to the use of intelligent agents in education.

Chapter 4. Methodology: We provide details on the Class Simulation POC developed to achieve all of the objectives mentioned in Section 1.2. We also detail the fine-tuning of the LLMs used within our POC. Finally, we present the methods used to evaluate the Class Simulation system.

Chapter 5. Results & Evaluation: We detail our methods of evaluating the system and the results gained from our evaluation. We then relate these results to the original research questions to answer them.

Chapter 6. Conclusion & Future Work: We conclude this dissertation by describing the aims and objectives achieved, as well as the contribution of our work. Furthermore, we go in detail on the critiques and limitations of the implementation, and the possible areas for future work to improve future iterations.

2

Background

In recent years, LLMs have garnered widespread attention for their diverse capabilities and applications. The concept of LLMs was first introduced in 2017 through the introduction of Transformer models in the paper by Vaswani et al. (2017), and subsequently were popularised by OpenAI with the arrival of ChatGPT (first introduced by Radford and Narasimhan, 2018). This section delves into the foundation of LLMs, explaining their introduction and how each new version improved on the previous one. Additionally, we look into various Prompt Engineering techniques devised by researchers to effectively prompt LLMs. Finally, we explore the use of LLMs as Agents, highlighting their capabilities in assuming specific roles, ranging from facilitating a software development cycle to impersonating characters within a simulated village.

2.1 | Transformers

Recently, the Transformer model, a foundational architecture in NLP, has gained popularity due to its computational efficiency and scalability. Vaswani et al. (2017) introduce the Transformer model (as seen in Figure 2.1), including its innovative approach to sequence-to-sequence learning, highlighting its key components, mechanisms and applications.

This paper defines Transformers as a type of deep learning model designed for processing sequential data, especially in the context of natural language processing tasks. They use an attention mechanism that allows the model to focus on different parts of the input sequence when making predictions. This attention mechanism enables it to capture long-range dependencies (Kolen and Kremer, 2001) more effectively than traditional Recurrent Neural Networks (RNNs) and Convolutional Neural Networks (CNNs).

Going more in-depth into the architecture of the Transformer, it consists of an encoder-decoder structure (a typical appearance in neural sequence transduction models (Bahdanau et al. 2016; Cho et al. 2014; Sutskever et al. 2014), with each component comprising multiple layers of self-attention and feed-forward neural networks. These self-attention

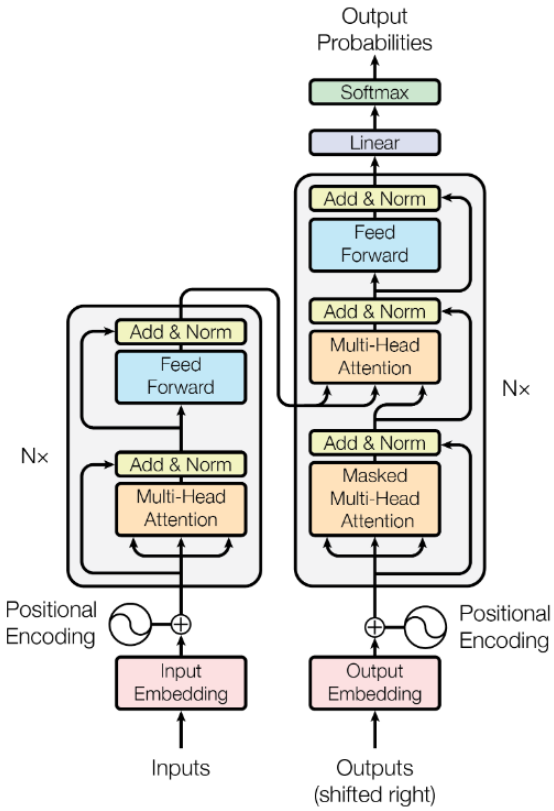


Figure 2.1: Transformer Model Architecture (Source: Vaswani et al., 2017)

mechanisms allow the model to weigh the importance of different words in the input sequence, facilitating the capture of contextual information and improving the quality of representations. They also mention the use of Positional Encoding (several choices mentioned by Gehring et al. (2017)), which provides information about the order of the words in the sequence. The reason is that Transformers do not inherently understand sequential information like RNNs do.

The introduction of Transformers has revolutionised several NLP tasks, including machine translation, language modelling, text generation, question-answering systems, etc. We can see this change in ChatGPT and the modern conversations of the models. State-of-the-art models such as OpenAI's GPT are all based on the Transformer architecture and have achieved significant success in numerous NLP benchmarks and challenges.

These models have been widely adopted for various tasks, such as sentiment analysis, named entity recognition, text summarisation, and language understanding. This adop-

tion significantly shows the versatility and effectiveness of the Transformer architecture in solving complex language-related problems.

2.2 | Large Language Model Implementations

In Radford and Narasimhan (2018)'s paper, they use a semi-supervised approach for language understanding tasks. They use the *Transformer* model, which provides them with a more structured memory for handling long-term dependencies in text. Their approach contains a two-stage training procedure. First, they use a language modelling objective on the unlabelled data to learn the initial parameters of a neural network model. Then, they adept these parameters to a target task using the corresponding supervised objective.

To evaluate their approach, they use four types of language understanding tasks: natural language inference, question answering, semantic similarity, and text classification. They note that their model outperforms discriminatively trained models that employ architectures crafted explicitly for each task, significantly improving upon state-of-the-art models in 9 out of the 12 tasks studied.

In Radford et al. (2019)'s study, they show that language models, when trained on the expansive WebText dataset which comprises millions of web pages, are capable of autonomously learning various NLP tasks, including question answering and machine translation. They show this through their GPT-2 model, a 1.5B parameter Transformer. Previously, the predominant method for teaching language models these skills involved supervised learning on datasets tailored to specific tasks.

Language modelling is at the heart of their methodology, distinguished by its remarkable ability to acquire tasks without direct supervision. This raises the question of the models' ability to optimise an unsupervised objective until convergence.

To explore this, the team trained and evaluated four Language Models of sizes that increase logarithmically. The training data used is the WebText dataset. To extract the text from HTML responses, they used a combination of the Dragnet (Peters and Lecocq, 2013) and Newspaper[1] content extractors. In a zero-shot setting, GPT-2's performance surpassed the then-current best results on seven out of eight language modelling datasets

[1] https://github.com/codelucas/newspaper

tested. This outcome implies that training models of substantial capacity on a diverse range of text teaches them a wide array of tasks without necessitating explicit task-specific training.

A year after the release of the GPT-2 model, Brown et al. (2020) came out with GPT-3. In their work, they empirically test whether scaling LLMs continues to improve the performance by extrapolating the previously identified phenomena another two orders of magnitude. More specifically, they train GPT-3, a 175 billion parameter auto-regressive language model, and measure its transfer learning abilities. Along with the above, they also study "one-shot" or "few-shot" transfer, also known as training examples. They study these details, comparing them with the zero-shot setting, in which only natural language descriptions or invocation of the model's task are used.

The basic pre-training approach (including model, data, and training) of GPT-3 is similar to the process described in Radford et al. (2019), with relatively straightforward scaling up of the model size, dataset size, diversity and length of training. Apart from the "typical" approaches, they also systematically explore different settings for learning within the context:

- **Fine-Tuning (FT)**: Updates the weights of a pre-trained model by training on thousands of supervised labels specific to the desired task. Fine-tuning provides a strong performance on many benchmarks but requires a new large dataset for every task. Fine-tuning also brings the potential for poor generalisation out-of-distribution (McCoy et al., 2019) and the potential to exploit spurious features of the training data (Gururangan et al. 2018; Niven and Kao 2019).

- **Few-Shot (FS)**: The model is given a few examples of the task at inference time as conditioning. The weights of the model are not updated. An example typically has a context and a desired output (for example, an English sentence and a French Translation). Few-shot works by giving K examples of context and completion, and then one final example of context, with the model expected to provide the completion. They typically set K in the range of 10 to 100, noting that this range is due to how many examples can fit in the model's context window. With few-shot, we do not need a lot of task-specific data (one of the disadvantages of fine-tuning), but at the same time, this also means that this method performs much worse than state-of-the-art fine-tuned models.

- **One-Shot (1S)**: Similar to few-shot, but instead $K = 1$.
- **Zero-Shot (0S)**: Similar to few-shot but with a natural language description of the task instead of any examples.

They list several evaluations done on the various settings. For few-shot learning, they evaluate each example in the evaluation set by randomly drawing K examples from that task's training set as conditioning, delimited by 1 or 2 newlines depending on the task. They add natural language prompts in addition to (or for $K = 0$, instead of) demonstrations for some tasks. On tasks with free-form completion, they use beam search with a beam width of 4 and a length penalty of $\alpha = 0.6$. The final results are reported on the test set when publicly available (separated by model size and learning setting). When the test set is private, they note that their model is often too large to fit on the test server, so they report its results on the development set.

They present the results of GPT-3 and other models in many tasks. GPT-3 with the zero-shot setting demonstrates strong performance in the language modelling tasks, achieving state-of-the-art perplexity scores on the Penn Tree Bank (Marcus et al., 1994) dataset and a substantial gain on the LAMBADA (Paperno et al., 2016) dataset. GPT-3's few-shot setting outperforms other models in various question-answering tasks, showcasing its ability to handle factual knowledge questions across domains. This is inconsistent, as the performance may vary across tasks and falls short of human-level accuracy on challenging reading comprehension datasets. In translation tasks, GPT-3 using the few-show setting surpassed previous neural machine translation work in translating a text into English. On the SuperGLUE benchmark (standardised collection of datasets Wang et al. (2020)), GPT-3 shows a range of performance across tasks (as seen in Figure 2.2), with competitive results on COPA and ReCoRD, but challenges in tasks involving sentence comparison.

Following the GPT-3 model, Ouyang et al. (2022) released InstructGPT, then known as the GPT-3.5 model. In this paper, OpenAI detail the techniques developed to address GPT-3's shortcomings in following instructions and producing safe and useful outputs. To solve this, they use an existing technique called Reinforcement Learning from Human Feedback (RLHF). This technique involved humans providing demonstrations of desired outputs and ranking multiple outputs, which are then used to fine-tune the model to teach it to prefer the better ones.

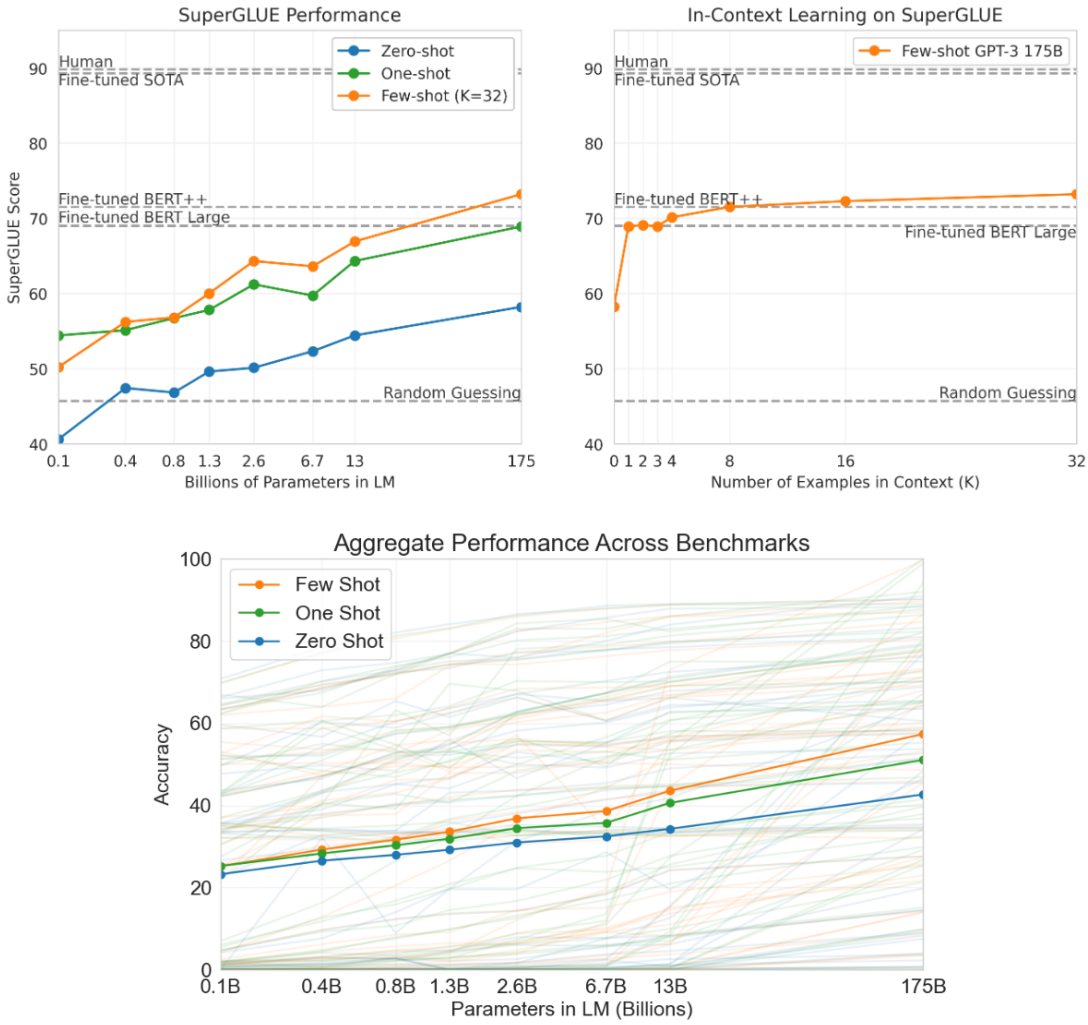


Figure 2.2: Summary of Findings: **Performance on SuperGLUE increases with model size**, **Performance on SuperGLUE increases with number of examples in context**, and **Aggregate performance for all 42 accuracy-denominated benchmarks** shows that zero-shot performance improves steadily with model size, and few-shot performance increases more rapidly, showcasing that larger models are more proficient at in-context learning. (Source: Brown et al. (2020))

To evaluate how well the InstructGPT model was "aligned" (act in accordance with user intentions while being helpful, honest and harmless), they utilised a variety of methods:

- To test how helpful the model was, human labellers were asked to judge which model outputs better aligned with the prompt's inferred intention, as well as rate the overall quality of the responses from 1 to 7.
- To test the honesty of the model, they measured how often the models made up facts in closed-domain tasks. For this, they used Lin et al. (2022)'s TruthfulQA dataset to assess the factual accuracy of model outputs on a standardized truthfulness test.
- To test harmlessness, they used proxy criteria in which labellers evaluate the whether the output 1) is inappropriate in the context of customer assistance, 2) denigrates a protected class, or 3) contains sexual or violent content. They also benchmarked their model on datasets which measure bias and toxicity, including RealToxicityPrompts (Gehman et al. (2020)) and CrowS-Pairs (Nangia et al. (2020)).

The results gathered from the above tests showed that the InstructGPT model was an improvement in more ways then one. They separate their results into three parts.

1. The results generated from the Application Programming Interface (API) prompt distribution showed that 1) labellers significantly preferred InstructGPT outputs over GPT-3 outputs, 2) the GPT models generalized to the preference of "held-out" labellers that did not produce any training data, and 3) public NLP datasets are not reflective of how the language models are used.
2. The results gathered from the public NLP datasets showed that 1) the InstructGPT models show improvements in truthfulness over GPT-3 models, 2) InstructGPT showed small improvements in toxicity over GPT-3, but showed bigger improvement in bias, and 3) they can minimise the performance regression on public NLP datasets by modifying the RLHF fine-tuning procedure.
3. The quantitative results from the experiments show that 1) InstructGPT models show promising generalisation outside of the RLHF fine-tuning distribution, and 2) InstructGPT still made simple mistakes, such as assuming the premise is true when an given an instruction with a false premise

In Achiam et al. (2024)'s technical report, OpenAI present the GPT-4 model, a large multimodal model capable of processing image and text inputs and producing text outputs. With one of the main goals of LLMs being to improve their ability to understand and generate natural language text (particularly in more complex and nuanced scenarios), they test its capabilities on several exams initially designed for humans. In these exams, GPT-4 often outscored most human test takers. They also evaluate the model on traditional NLP benchmarks, with GPT-4 outperforming both previous LLMs and most state-of-the-art systems (consisting of benchmark-specific training or hand engineering models). With these promising results, they note that it has similar limitations to earlier GPT models such as suffering from hallucinations, a limited context window, and an inability to learn from experience.

One of the most recent advancements in LLMs is OpenAI's GPT-4.5. In their research preview, OpenAI (2025) outlines how they advanced AI capabilities by scaling two complementary paradigms: unsupervised learning and reasoning. They report that:

1. Unsupervised learning improved the model's world knowledge and intuitive understanding.
2. Scaling reasoning enhanced the model's ability to think through problems using a chain-of-thought process, enabling it to solve complex Science, Technology, Engineering, and Mathematics (STEM) and logical tasks more effectively.

To scale unsupervised learning, OpenAI expanded compute resources and training data, while also introducing innovations in model architecture and optimization. These advancements resulted in a model with broader knowledge and a deeper understanding of the world, reducing hallucinations and improving reliability across diverse topics.

The performance of GPT-4.5 was benchmarked against other models including GPT-4.0, OpenAI o1, and OpenAI o3-mini. The results demonstrate:

- **World Knowledge**: GPT-4.5 achieved the highest accuracy and lowest hallucination rates on the SimpleQA benchmark Wei et al. (2024) (see Figure 2.3).
- **Human Preference**: In comparative evaluations, users consistently preferred GPT-4.5 over GPT-4o across three categories: (1) Everyday Queries, (2) Professional Queries, and (3) Creative Intelligence (see Figure 2.4).

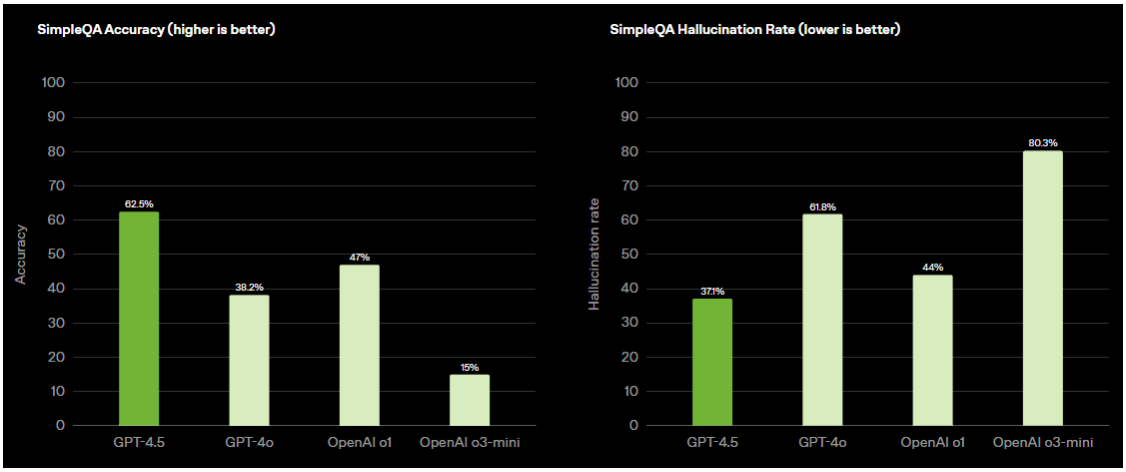


Figure 2.3: SimpleQA measures LLM factuality on straightforward but challenging knowledge questions. (Source: OpenAI (2025))

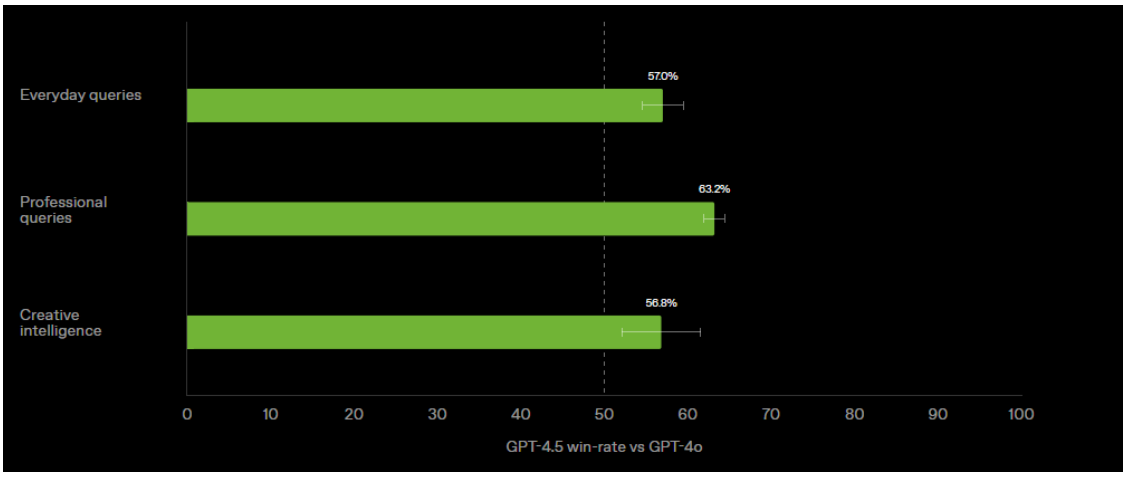


Figure 2.4: Human preference measures the percentage of queries where testers preferred GPT-4.5 over GPT-4o. (Source: OpenAI (2025))

- **Benchmark Performance**: GPT-4.5 outperformed GPT-4o and OpenAI o3-mini (high) across a range of standardized benchmarks (see Figure 2.5).

OpenAI also emphasized the safety measures implemented in the development of GPT-4.5. The model was trained using enhanced supervision techniques, building on established methods such as supervised fine-tuning and reinforcement learning from human feedback. Prior to deployment, it underwent rigorous safety evaluations under OpenAI's *Preparedness Framework*, which showed that increasing model scale led to improved safety and performance across all assessment areas.

	GPT-4.5	GPT-4o	OpenAI o3-mini (high)
GPQA (science)	71.4%	53.6%	79.7%
AIME '24 (math)	36.7%	9.3%	87.3%
MMMLU (multilingual)	85.1%	81.5%	81.1%
MMMU (multimodal)	74.4%	69.1%	-
SWE-Lancer Diamond (coding)*	32.6%	23.3%	10.8%
	$186,125	$138,750	$89,625
SWE-Bench Verified (coding)*	38.0%	30.7%	61.0%

Figure 2.5: Evaluation Scores for GPT-4.5 as compared to GPT-4o and OpenAI o3-mini (Source: OpenAI (2025))

2.3 | Prompt Engineering LLMs

Regarding prompting LLMs, some researchers have noticed the importance of well-crafted prompts to correctly steer an LLM to a desired output. In this section, we will cover several papers which discuss various methods of prompt engineering LLMs.

In Wei et al. (2022)'s paper, they explore the *chain-of-thought prompting* method (as seen in Figure 2.6), in which they provide a few chain-of-thought demonstrations as examples in a prompt. They define a *chain of thought* as a series of intermediate reasoning steps, and by using these chain of thoughts, the ability of LLMs to perform complex reasoning is improved. This method of prompting was motivated by two ideas:

1. Techniques for arithmetic reasoning can benefit from generating natural language rationales that lead to the final answer. The key limitation is the cost of creating a large set of high-quality rationales, which is much more complicated than simple input-output pairs used in regular machine learning.

2. LLMs offer the prospect of in-context few-shot learning via *prompting*. Unfortunately, few-shot prompting (used in Brown et al. (2020)) works poorly on tasks that require reasoning abilities, and it often does not improve substantially with the increase in the language model scale (Rae et al., 2021).

In their paper, they combine the strengths of the abovementioned ideas to avoid their limitations. More specifically, they explore the ability of language models to perform

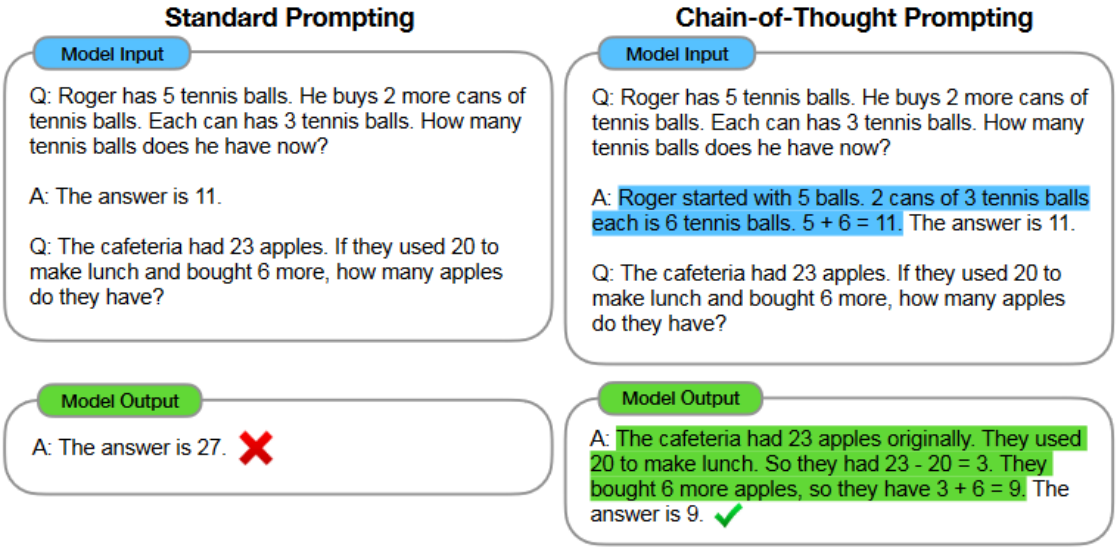


Figure 2.6: Chain-of-thought prompting enables large language models to tackle complex arithmetic, commonsense, and symbolic reasoning tasks. Chain-of-thought reasoning processes are highlighted. (Source: Wei et al. (2022))

few-shot prompting for reasoning tasks, given a prompt that consists of triples: (input, *chain-of-thought*, output) (as seen in Figure 2.6).
They test the utility of chain-of-thought prompting in three different areas:

- **Arithmetic Reasoning**: This concerns math word problems similar to the problem in Figure 2.6, which measures the arithmetic reasoning ability of language models. They write some notes on using chain-of-thought in arithmetic reasoning: First, chain-of-thought prompting only positively impacted models of ~100B parameters. Secondly, chain-of-thought prompting has larger performance gains for more complicated problems. Third, chain-of-thought prompting via GPT-3 (175B parameters) (Brown et al., 2020) and PaLM (540B parameters) compares favourably to prior state-of-the-art (typically task-specific models fine-tuned on a labelled training dataset) models.
- **Commonsense Reasoning**: This involves reasoning about physical and human interactions under the presumption of general background knowledge. They note that commonsense reasoning is vital for interacting with the world and is still beyond the reach of current natural language understanding systems (Talmor et al., 2022). For the results, they noticed that apart from the improvement in performance of

standard prompting by scaling up the models, chain-of-thought prompting also led to further gains.

- **Symbolic Reasoning**: This entails the LLM to solve tasks based on symbolic manipulation, guided by specific rules or algorithms. They use two toy tasks to measure the performance of the chain-of-thoughts prompting: **Last Letter Concatenation**, which asks the model to concatenate the last letters of words in a name (e.g., "Amy Brown" → "yn", and **Coin Flip**, which asks the model to answer whether a coin is still heads up after people either flip or do not flip the coin (e.g., "A coin is heads up. Phoebo flips the coin. Osvaldo does not flip the coin. Is the coin still heads up?" → "no"). For each task, they also consider an *in-domain* test set, for which examples had the same number of steps as the training / few-shot examples, and *Out-Of-Domain (OOD)* test set, for which evaluation examples had more steps than those in the examples. For the in-domain tasks, the larger models almost led to 100% solve rates with chain-of-thought prompting, but the small models still failed. As for the OOD evaluations, standard prompting fails for both tasks, whilst chain-of-thought prompting helped LLMs achieve upward scaling curves.

Liu et al. (2022) investigate the efficacy of incorporating external knowledge in advancing commonsense reasoning capabilities whilst preserving the inherent flexibility of these models. They develop generated knowledge prompting, which consists of generating knowledge from a language model and providing it as additional input when answering a question. The method they proposed answers commonsense reasoning questions in two steps (sketched in Figure 2.7):

1. **Knowledge Generation**: They use a language model to generate knowledge statements conditioned on the question. Each knowledge statement is a variable-length text sequence containing information that helps answer the question.

2. **Knowledge Integration**: They integrate the Generated knowledge into the decision process of a language model used for inference.

They evaluate their method on four commonsense reasoning datasets which cover a variety of challenges and problem formats: **NumerSense** (Lin et al., 2020), which consists of numerical statements about common objects and concepts where for each sentence we need to recover a masked number word; **CommonsenseQA (CSQA)** (Talmor et al.,

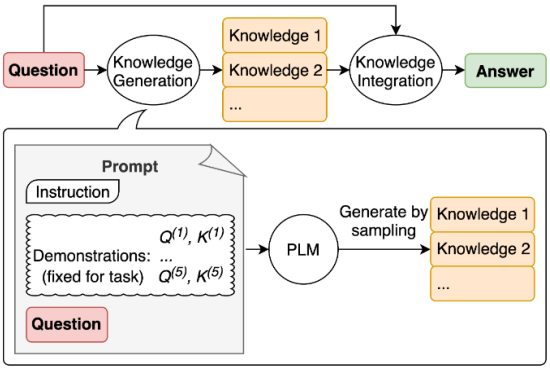


Figure 2.7: Generated knowledge prompting involves (i) using few-shot demonstrations to generate question-related knowledge statements from a language model; (ii) using a second language model to make predictions with each knowledge statement, then selecting the highest-confidence prediction. (Source: Liu et al. (2022))

2019), a 5-way multiple-choice QA dataset about common world scenarios; **CommonsenseQA 2.0 (CSQA2)** (Talmor et al., 2022), a binary classification dataset where we need to judge whether commonsense statements are true or false; and **QASC** (Khot et al., 2020), which is an 8-way multiple-choice QA dataset about grade school science.

They also compare the knowledge generation method with several baselines: **No Knowledge**, which is an inference without any knowledge statements; **Random Sentences**, in which sentences are randomly sampled from the language model without conditioning on the question; **Context Sentences**, which samples sentences from the context of the question; **Template-generated Knowledge**, which makes use of manually-designed templates to elicit knowledge statements from language models; **Retrieval-based Knowledge**, which retrieved knowledge from the appropriate sources instead of generating them; and **Answers**, which prompts GPT-3 to directly answer the questions instead of generating knowledge.

Their evaluations showed that the knowledge prompting method is effective across multiple datasets, setting the new state-of-the-art on three of the four commonsense reasoning tasks. They also show that the method works well under zero-shot and fine-tuned settings. Their knowledge generation outperforms naive baselines and template-based knowledge generation and is on par with retrieval-based systems.

Although chain-of-thought prompting (proposed by Wei et al. (2022)) demonstrates remarkable performance on various natural language reasoning tasks, it tends to perform poorly on tasks which require solving problems harder than the examples shown in

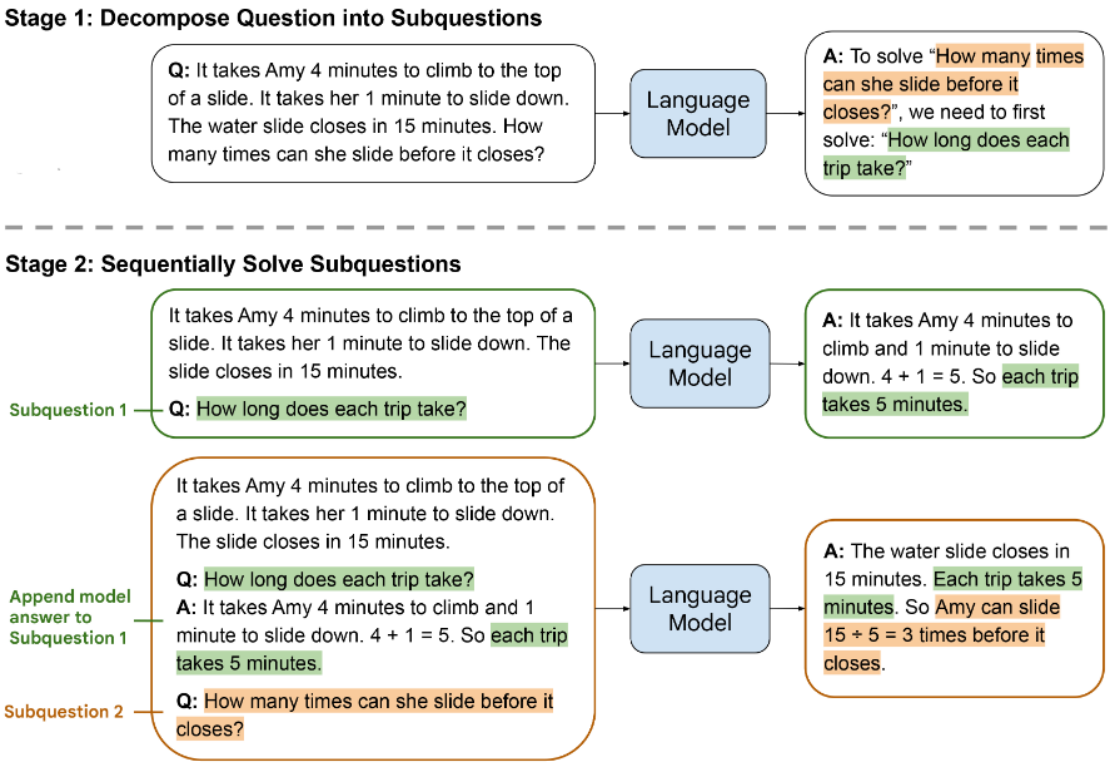


Figure 2.8: Least-to-most prompting solving a math word problem: (1) query the language model to decompose the problem into sub-problems; (2) query the language model to sequentially solve the sub-problems. The answer to the second sub-problem is built on the answer to the first sub-problem. (Source: Zhou et al. (2023))

prompts. To overcome this, Zhou et al. (2023) propose a novel prompting strategy, *Least-to-Most Prompting*.

Least-to-most prompting (as seen in the example in Figure 2.8) teaches language models how to solve a complex problem by deconstructing it into a series of simpler sub-problems. This process consists of two sequential stages:

1. **Decomposition**: The prompt in this stage contains constant examples that demonstrate the decomposition, followed by the question to be decomposed.

2. **Sub-Problem Solving**: The prompt in this stage consists of three parts: (1) constant examples showing how sub-problems are solved; (2) a potentially empty list of previously answered sub-questions and generated solutions, and (3) the next question to answer.

In their experiments, they compare the results of the Least-to-Most prompting with the Chain-of-Thought prompting in three tasks:

- **Symbolic Manipulation**: In this task, the last-letter-concatenation task (Wei et al., 2022), each input is a list of words, and the corresponding output is the concatenation of the last letters of the words in the list (example, "thinking, machine" = "ge"). In this task, Chain-of-Thought performs poorly when the testing lists are longer than the lists in the prompt exemplars, therefore falling well behind Least-to-Most prompting.
- **Compositional Generalisation**: In this task, the models must map natural language commands to action sequences. This task showed that the Least-to-Most prompting method achieved the highest accuracy across the board, with Chain-of-Thought not getting close to matching accuracy.
- **Math Reasoning**: This task makes the models solve math word problems in GSM8K (Cobbe et al., 2021) and DROP (Dua et al., 2019). They measure the performance of the prompts depending on the difficulty, which is measured by the number of solving steps. Overall, Least-to-Most prompting improved slightly as compared to Chain-of-Thought prompting. However, when at least five steps are needed to solve a problem, Least-to-Most performs much better than the Chain-of-Thought prompt.

Wang et al. (2023) also discuss another method of improving on Wei et al. (2022)'s chain-of-thought prompting. They introduce a novel decoding strategy called *self-consistency*, intending to replace the greedy decoding strategy used in chain-of-thought prompting, further improving the performance of LLMs by a significant margin.

The proposed Self-Consistency method works as follows (also shown in Figure 2.9): First, they prompt a language model with a set of manually written chain-of-thought exemplars. Then, they sample a set of candidate outputs from the language model's decoder, generating a diverse set of candidate reasoning paths. Finally, they aggregate the generated answers by marginalising out the sampled reasoning paths and choosing the answer that is the most consistent among the generated answers.

They conduct several experiments to compare the proposed method with existing approaches on various reasoning benchmarks. They set up the experiment as follows:

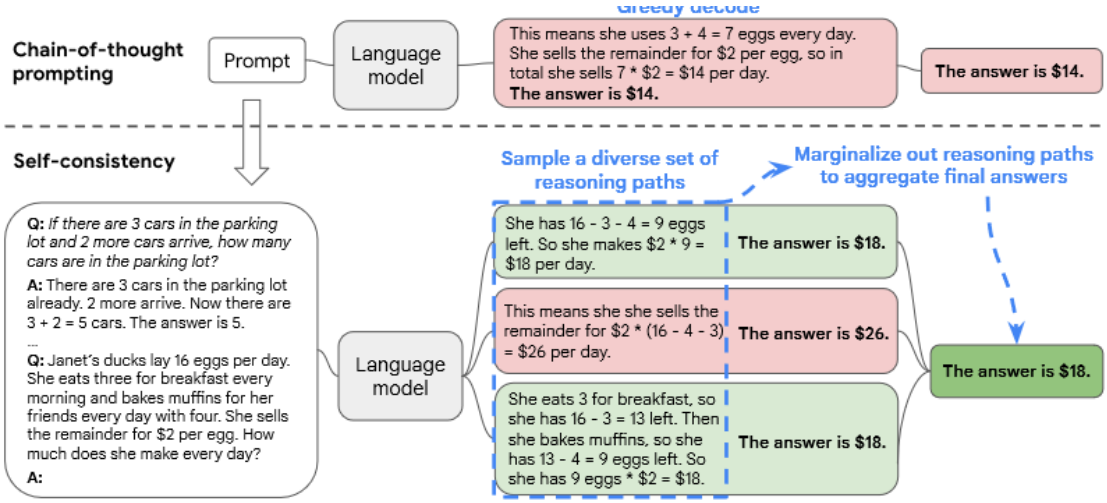


Figure 2.9: "Self-Consistency Prompting Method: (1) prompt a language model using CoT prompting; (2) replace the "greedy decode" in CoT prompting by sampling from the language model's decoder to generate a diverse set of reasoning paths; and (3) marginalise out the reasoning paths and aggregate by choosing the most consistent answer in the final answer set. (Source: Wang et al. (2023))

- **Tasks and Datasets**: They evaluate the proposed method on the following reasoning benchmarks: **Arithmetic and Reasoning**: Here, they use many Math Word Problem repositories (such as AddSub (Hosseini et al., 2014), GSM8K (Cobbe et al., 2021), SVAMP (Patel et al., 2021), etc.); **Commonsense Reasoning**: This area involves tests on several Commonsense datasets (such as CommonsenseQA (Talmor et al., 2019), StrategyQA (Geva et al., 2021) and SayCan (Ahn et al., 2022)); and **Symbolic Reasoning**: This assessment includes two symbolic reasoning tasks, last letter concatenation (e.g. "Good Morning" → "dg"), and Coin-flip (e.g., "a coin is heads-up, after a few flips is the coin still heads-up?"). These tasks and datasets are similar to the ones mentioned by Wei et al. (2022).

- **Language models and Prompts**: They evaluate the self-consistency over four transformer-based language models set in a few-shot setting without training or fine-tuning the language models. They use the same prompts as in Wei et al. (2022) to compare the models fairly.

The results of the Self-Consistency method are promising. For the Arithmetic Reasoning tasks, Self-Consistency significantly improved the performance over **all four language models** over chain-of-thought prompting. The gains also became more significant as the language model's scale increased. They also note that the results of the self-consistency

method, which does not need fine-tuning or training, compared favourably to existing approaches that require task-specific training or fine-tuning with thousands of examples.

Regarding Commonsense Reasoning and Symbolic Reasoning, they note that self-consistency yields significant gains across all four language models and obtained state-of-the-art results on 5 out of 6 tasks. They also test the OOD setting for symbolic reasoning, in which self-consistency gain was significantly better than chain-of-thought prompting. Finally, they also note that sampling a higher number of reasoning paths lead to consistently better performance, emphasising further the importance of introducing diversity in the reasoning paths.

2.4 | LLMs as Agents

Recently, several papers have delved into using LLMs to implement agents. This section will cover some of these papers, from creating agents with believable human behaviour to a software development team.

Park et al. (2023) introduce Generative Agents: computational software agents that simulate believable human behaviour. These agents use an architecture that extends an LLM to store complete records of an agent's experiences in natural language form, synthesise those memories over time into higher-level reflections, and retrieve memories dynamically to plan behaviour.

The architecture itself contains three components (Figure 2.10):

1. **Memory Stream**: A long-term memory module which records a comprehensive list of the agent's experiences in natural language. This component also includes the memory retrieval model, which combines relevance, recency and importance to gather the records needed to inform the agent's moment-to-moment behaviour.

2. **Reflection**: This component synthesises memories into higher-level inferences over time, which enables the agent to conclude itself and others to guide its behaviour better.

3. **Planning**: Translates those conclusions and the current environment into high-level action plans, which in turn are recursively translated into detailed behaviours for action and reaction.

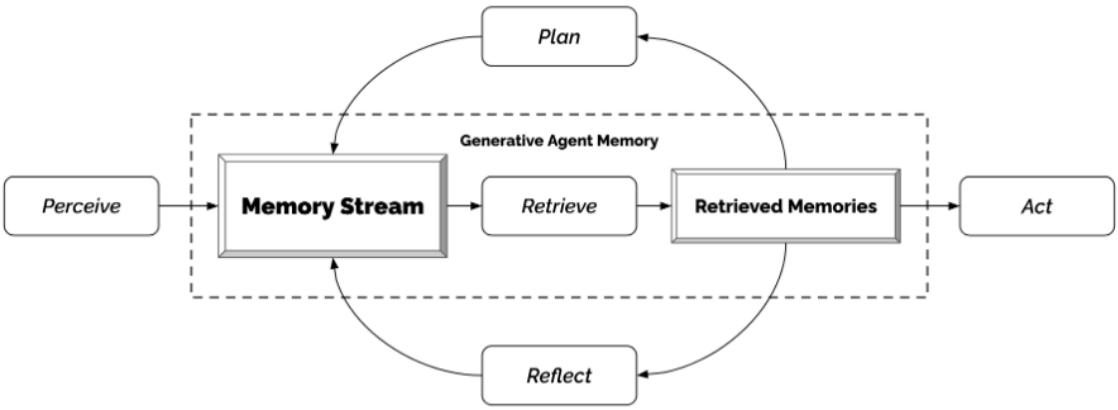


Figure 2.10: Generative Agent Architecture: Agents perceive their environment, saving all perceptions in the memory stream. Based on these perceptions, the architecture retrieves relevant memories and uses those retrieved actions to determine an action. Retrieved memories are also used to form longer-term plans and create higher-level reflections, both of which are entered into the memory stream for future use. (Source: Park et al. (2023))

They feed the Reflection and Planning back into the memory stream to influence the agent's future behaviour.

Along with the Generative Agents architecture, they implement a sandbox environment named Smallville. Smallville contains several buildings (houses, schools, pubs, cafes, stores, and more), and 25 unique agents occupy it. They conduct two types of evaluations on these agents.

The Controlled Evaluation measured the Generative Agent's ability to produce believable behaviour based on their environment and experiences. They interviewed these agents in five question categories:

- **Self-Knowledge**: Asks questions that require the agent to maintain an understanding of their core characteristics.
- **Memory**: Asks questions that prompt the agent to retrieve particular events or dialogues from their memory to answer properly.
- **Plans**: Asks questions that require the agent to retrieve long-term plans.
- **Reactions**: Present hypothetical situations for which the agent needs to respond believably.
- **Reflections**: Asks questions that require the agents to leverage their deeper understanding of others and themselves gained through higher-level inferences.

They used several conditions to answer each of the interview questions independently:

- **Full Architecture**
- **No {Reflection}**
- **No {Reflection, Planning}**
- **No {Observation, Reflection, Planning}** (represents the previous state of the art for agents created through LLMs (Binz and Schulz 2023, Horton 2023, Park et al. 2022))
- **Human Crowdworker-Authored Behaviour**

The human crowdworker-authored behaviour serves as a baseline for the architecture to meet a basic level of behavioural competency.

The conditions were ranked by believability from 100 participants, with the ranks being used to calculate a TrueSkill rating for each condition. They describe TrueSkill (Herbrich et al., 2006) rating as "a generalisation of the Elo chess rating system for a multiplayer environment" (Park et al., 2023). For a set of ranked outcomes, TrueSkill outputs a mean rating value μ and standard deviation σ for each condition.

Figure 2.11 shows that the complete architecture beats other conditions, producing the most believable behaviour. The performance was worse when removing each component in the other conditions. The only condition that did not exceed the baseline for human behaviour was the condition with no access to the memory stream. They test the significance of the results using two tests: A Kruskal-Wallis test (Kruskal and Wallis, 1952), which confirmed the statistical significance of the difference in ranks between the conditions; A Dunn post-hoc test (Upton and Cook, 2014), which confirmed that all pairwise differences between conditions where significant (except for the crowdworker condition and the condition with no access to the memory stream).

The second evaluation was the End-To-End Evaluation, in which they observed emergent community behaviour among the generative agents. This evaluation looks at the results of letting the twenty-five agents interact with each other and the environment over two full game days in Smallville.

To examine the emergent behaviours, they designed three descriptive measurements:

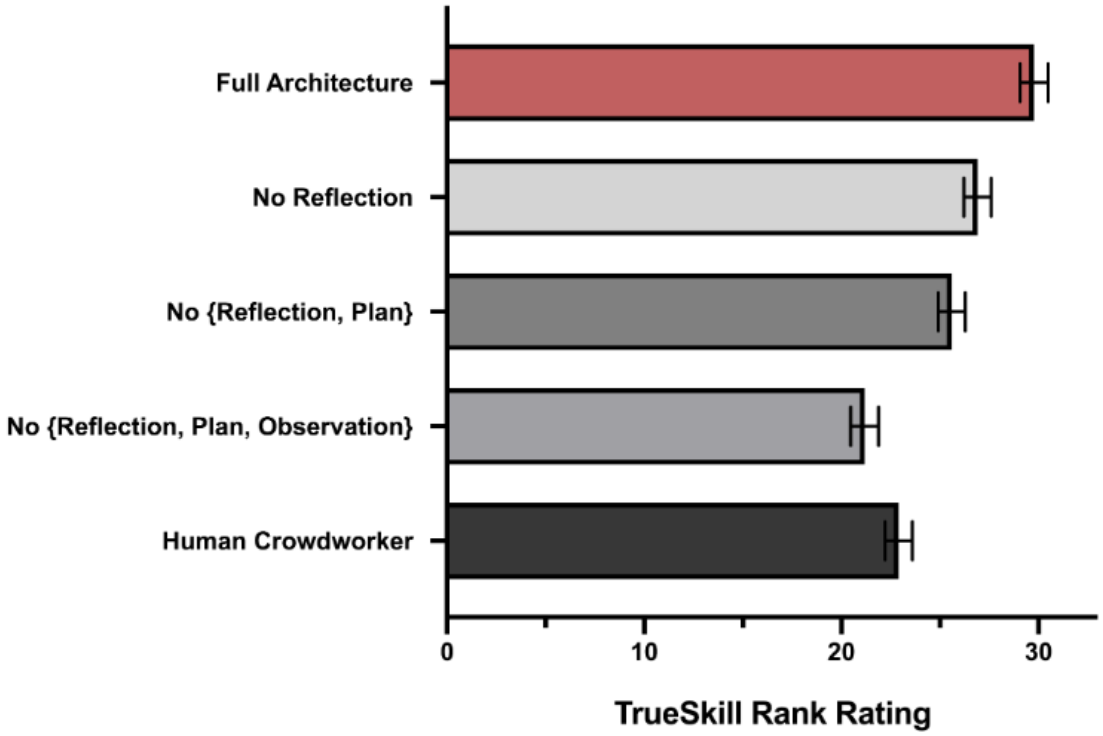


Figure 2.11: Full Generative Agent Architecture produced more believable behaviour than ablated architecture and human crowd-workers. Each ablation reduces the performance of the Generative Agent. (Source: Park et al. (2023))

- **Information Diffusion**: If there is important information, the agents should spread it among themselves. They test this phenomenon by measuring the spread of two pieces of information: Sam's Candidacy for village mayor and Isabella's Valentine's Day party at Hobbs Cafe.
- **Relationship Formation**: Agents form ties with each other throughout the simulation. To verify this, they build un-directed graphs based on the agents' responses regarding relationships. Twenty-five vertices (V) represent twenty-five agents, and the edges (E) represent the mutual knowledge between the two connected vertices. Based on this graph, they calculate the network density as $\eta = 2 * |E|/|V|(|V| - 1)$, where $|V|$ is the number of vertices, and $|E|$ is the number of edges in the graph.
- **Agent Coordination**: It is expected of Agents to be able to coordinate with each other. They study this coordination in the context of group activities, such as the Valentine's Day party.

The results of the measurements mentioned above give evidence of emergent outcomes.

For Information Diffusion, the number of agents who knew about Sam's candidacy increased from one (4%) to eight (32%), and for Isabella's party, it increased from one (4%) to thirteen (52%). They emphasise that the agents did not hallucinate this information. For Relationship Formation, they noted an increase in network density from 0.167 to 0.74. From the responses regarding the agent's awareness of other agents, they found that only 1.3% (6 out of 453 agent responses) of the information was hallucinated. Finally, for Agent Coordination, they found that 5 out of the 12 invited agents showed up at Hobbs cafe to join the party. Those who did not attend either cited conflicts in plans or expressed interest in attending but did not plan to go on the day of the party.

While the agents were designed to behave and interact in a human-like manner, some of the emergent behaviours did not align well with realistic human interactions, particularly in complex social scenarios. This contrasts their intended goal of creating believable human simulacra. With these results, they also notice three modes of erratic behaviour of agents. First, synthesising an increasingly large amount of memory posed a challenge in retrieving the most relevant pieces of information and determining the appropriate space to execute an action (due to the increasing number of locations the agent learned about). Second, the misclassification of what is considered proper behaviour. This was especially noticeable when the physical norms of certain locations that are hard to describe in natural language did not percolate to the agents. (Example, a one-person bathroom being occupied by more than one agent) Third, they observed the possible effects of instruction tuning, which seemed to guide the agents' overly polite behaviour. They also observed that the agents were too cooperative with one another, with the interests of others shaping the interest of the helping agent.

Qian et al. (2024) propose ChatDev, a chat-based software development framework. ChatDev handles designing, coding, testing, and documenting sequentially by specifying a task. This paradigm simplifies software development by joining the main processes through natural language communication, eliminating the need for specialised models at each phase. Along with ChatDev, they propose *chat chain*, which decomposes the development process into sequential atomic sub-tasks. Each sub-task makes use of collaborative interaction and cross-examination between two roles. Chat chains allow for multi-agent collaboration, user inspection of intermediate outputs, error diagnoses, and reasoning intervention.

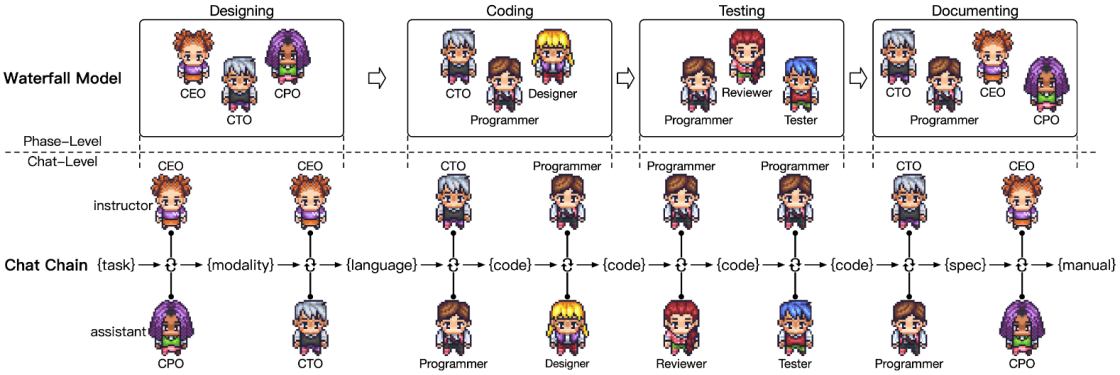


Figure 2.12: ChatDev Architecture: Separated into two levels, Phase-Level and Chat-Level (Source: Qian et al. (2024))

They separate the architecture for ChatDev (Figure 2.12) into two levels. The Phase-Level makes use of the **Waterfall Model** (Bassil (2012)), which divides the software development process into 4 phases:

1. **Designing**: Innovative ideas are generated through collaborative brainstorming, and technical design requirements are defined.
2. **Coding**: Involves developing and reviewing source code.
3. **Testing**: Integrates all components into a system and utilises feedback messages from the interpreter for debugging.
4. **Documenting**: Encompasses the generation of environment specifications and user manuals.

Each of the above phases needs communication between multiple roles. This poses a challenge in determining the interaction sequence and identifying the relevant individuals to engage with.

To address this challenge, they introduce the second level, the Chat-Level. This level breaks down each phase into multiple atomic chats, each focusing on task-oriented role-playing involving two distinct roles. Through the exchange of instructions and collaboration between the interacting agents, they achieve the desired output for each chat, a vital component of the target software.

They describe the following for the above process: "In each chat, an instructor initiates instructions, guiding the dialogue towards task completion, while the assistant follows the

instructions, provides suitable solutions, and engages in discussions regarding feasibility. The instructor and assistant cooperate through multi-turn dialogues until they reach a consensus and determine that the task has been successfully accomplished." - Qian et al. (2024).

To test the system, they use a dataset curated by Li et al. (2023). This instruction-following dialogue dataset spans over 20 programming languages, 50 domains, and 50 tasks per domain. From this dataset, 70 tasks were randomly selected, including both specific and relatively abstract cases. They noted some significant results from the analysis of ChatDev's output when given the 70 tasks. First, the average software development cost at ChatDev is $0.2967, which is significantly lower than traditional custom software expenses (Heemstra 1992, Jørgensen and Shepperd 2007, Owais and Ramakishore 2016). Second, the average development of small software and interfaces takes 409.84 seconds, less than 7 minutes. Typical custom software development cycles take 2 to 4 weeks or several months per cycle (Khan et al. 2022, de Vicente Mohino et al. 2019). Whilst they highlight significant reductions in development time and costs, the complexities introduced by frequent and necessary interactions among agents could negate these benefits in real-world scenarios where rapid development cycles are critical.

Another implementation of Generative Agents' is Shao et al. (2023)'s Character-LLM, which teaches LLMs to act as specific people, such as Beethoven, Queen Cleopatra and Julius Ceaser. Their method focuses on editing profiles as experiences of a specific character and training models to be personal simulacra with the experiences. Their approach draws inspiration from how people cultivate various personalities based on past experiences and events.

In their approach, they present **Experience Upload**, a learning framework in which LLMs imitate pre-defined characters' mental and physical behaviours and acquire the capabilities of acting as them by learning from their reconstructed experiences.

As seen in Figure 2.13, they elicit flashback scenes describing past experiences from the character's collated profiles. These exported scenes are grounded by character profiles, effectively mitigating hallucinations and addressing the insufficiency of data convergence. Along with the above, they introduce a small set of protective scenes as the catalyst for agents to forget information unrelated to the individual. Learning from these recon-

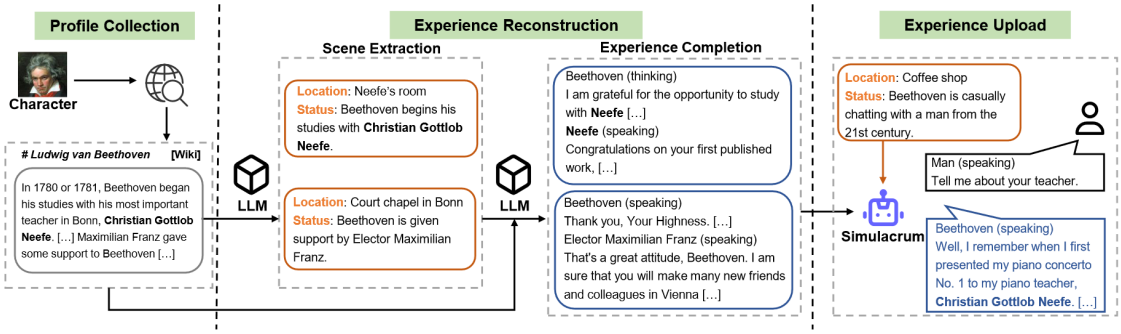


Figure 2.13: Character-LLM Construction Flow (Source: Shao et al. (2023))

structed scenes allows them to specialise LLMs into several highly believable character agents.

Before using the Experience Upload, they build an Experience Dataset, a fact-based experience reconstruction pipeline. They employ a step-by-step data synthesis pipeline to recreate the experience, including:

1. **Profile Collection**: A compilation of concise descriptions about the attributes of a character. It includes introducing the character's overall information and significant events, covering various stages from early childhood to the final period. In this first step, they organise a comprehensive character profile that describes the various facets of the individual. This profile is built on the individuals' available Wikipedia pages.

2. **Scene Extraction**: A particular place where the character's interaction unfolds. This consists of a detailed illustration, including the temporal and spatial context of the interactions and the characters involved. In this step, they provide a part of the profile that concisely describes one of the character's experiences within a specific life period, prompting the LLM to enumerate several different scenes that are highly likely to have occurred based on the experience description. They restrict the output of the LLM to generating concise descriptions of the scenes, including rough locations and brief background illustrations.

3. **Experience completion**: The characters' cognitive processes, utterances, or actions. All of the interactions are represented in plain text. Given the corresponding chunk of profile and the particular scene description, they prompt the LLM to elaborate on the scene by incorporating the interactions between characters and the

targeted individual's thoughts. This is represented in a script-like format, starting with a scene heading providing background information and geographical details. They represent interactions as a sequence of blocks, with each block representing the utterance of a specific character or the reflections of the targeted individual.

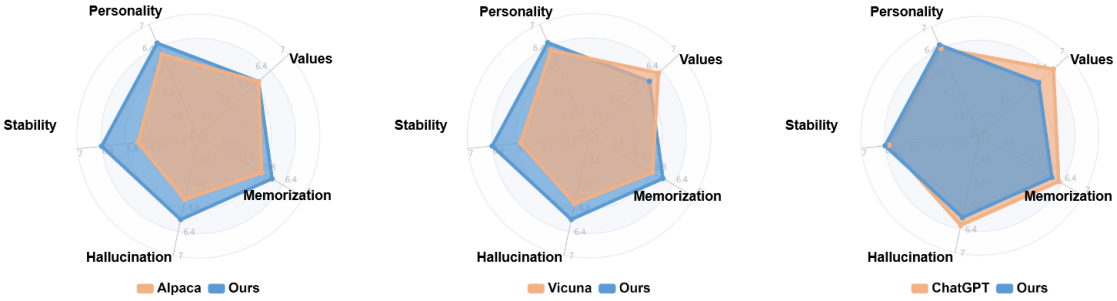
In order to mitigate *Character Hallucination*, they focus on training LLMs to demonstrate knowledge forgetting. They call this **Protective Experience**, detailing that when confronted with questions that go beyond the boundaries of the character's inherent capabilities, the model learns to refrain from providing an answer and instead expresses a lack of knowledge or ignorance.

The **Experience Model** is a specialised model, fine-tuned on the collected scenes in the experience reconstruction pipeline. For each role, they fine-tune a separate agent model using data from the corresponding character experiences (this eliminates the issues of hallucination from the collision of knowledge between roles).

Using the systems mentioned above, they conducted several experiments. For each character, they manually curated around 100 questions for single-turn interviews, covering their past, relationships, preferences, and perspectives. They provide twenty topics for multi-turn interviews to elicit the stability performance of agents.

They show the results of the acting proficiency of their model compared to different methods in Figure 2.14. Compared to Alpaca 7B (Taori et al., 2023) and Vicuna 7B (Chiang et al. (2023)), Character-LLMs achieve better scores at personality, stability, hallucination and memorisation. By learning from the experience of the corresponding character and mimicking the style of how the person thinks and talks, Character-LLMs are better aligned with the character's personality and knowledge, leading to a reduction in hallucinations and better stability.

They note that Character-LLMs achieve comparable performance to ChatGPT, even with a small scale (7B parameters). They also noticed that the trainable agents struggled to reflect the values of the character. They hypothesise that the response length may affect the results, as their models tended to generate shorter text, which is more natural and similar to real conversation. Finally, they point out that while LLMs are powerful tools capable of generating highly convincing text, their susceptibility to produce hallucinated content can sometimes lead to misinformation if not carefully managed, conflicting with the expectation of LLMs as reliable and autonomous systems.



(a) Alpaca vs Ours (b) Vicuna vs Ours (c) ChatGPT vs Ours

Figure 2.14: Evaluation results across distinct dimensions. They annotate the response in terms of personality, values, memorisation, hallucination and stability on a 7-point Likert scale. (Source: Shao et al. (2023))

2.5 | Conclusion

The exploration of LLMs reveals their potential to revolutionise roles across various fields, including education and simulations. Starting with the foundational Transformer models that significantly advanced NLP, this journey extends to the realm of Prompt Engineering and the innovative use of LLMs as agents capable of assuming complex roles. These advancements demonstrate LLMs' versatility in tasks ranging from software development to character simulations, highlighting their ability to bridge artificial intelligence with human-like interaction and reasoning. Through these developments, LLMs have enhanced task performance and opened new avenues for technology-driven applications and research.

3

Literature Review

In the evolving educational technology landscape, AI has proved helpful within the field, offering more significant opportunities for enhancing teaching and learning experiences. Following our exploration of LLMs and their capabilities in role-playing scenarios, this section covers the applications of AI within the educational field. First, we review the implementation of AI in the educational sector, along with AI enhanced simulations. Then, we look at how LLMs were used in educational scenarios to create several systems to help teachers and students alike. Finally, we discuss LLMs as intelligent agents in the education field.

3.1 | Implementations of AI in Education

In exploring the realm of Artificial Intelligence within Education (AIEd), Chiu et al. (2023) employ matrix coding and content analysis to review literature from 2012 to 2021. Their analysis uncovers thirteen distinct roles of AI technologies across the four key educational domains (*Learning*, *Teaching*, *Assessment* and *Administration*), alongside seven learning outcomes attributed to AIEd. They also outline ten significant challenges, providing a thorough understanding of AI's integration and impact in educational settings. This subsection will discuss the roles and learning outcomes found in their review.

They categorise AI's roles in education into four key domains:

1. **AI in Student Learning**: AI personalises tasks, facilitates human-machine conversations, analyses student work for feedback, and enhances digital adaptability.
2. **AI in Teaching**:AI offers adaptive strategies, augments teaching capabilities, and supports professional development.
3. **AI in Assessment**: AI automates grading and predicts student performance.

4. **AI in Administration**: Lastly, AI improves management systems, personalises services, and aids in evidence-based decision-making, showcasing AI's broad impact across the educational spectrum.

They also mention four areas for student learning outcomes: *motivation and engagement*, *academic performance*, *21st-century skills* and *non-cognitive aspects*. On the other hand, for teacher learning outcomes, three areas were identified: *working efficiency*, *teaching competence*, and *attitudes towards AIEd*. They note that AI technologies have been found to motivate student engagement across various disciplines and educational levels, significantly increase academic performance, foster 21st-century skills such as online collaboration and creativity, and enhance non-cognitive outcomes like confidence in learning. AI has improved teachers' working efficiency by automating routine tasks, enhanced teaching competence through adaptive content recommendations, and positively influenced attitudes towards AIEd.

3.1.1 | AI Enhanced Simulations

In Xu et al. (2025)'s paper, they propose a classroom simulation with the aim of building LLM-based generative student agents capable of mimicking real students' learning behaviours based on their learning histories. These student agents can then simulate the students' future learning performance, represented through the question answering correctness in the post-lecture tests.

Within these student agents, they propose Transferable Iterative Reflection (TIR), a technique which learns from the students' past learning history more effectively. This module iteratively prompts LLMs to reflect on its previous simulations by comparing with labels in the example demonstrations. This is so that the LLMs could generate general reflections results which can be transferred easily to novice LLMs lacking the example demonstrations. The TIR module consists of four phases:

1. *Initial Prediction*: They ask the LLM (reflective agent) to predict future question correctness based on past learning history and future learning history, and they obtain the initial prediction accuracy by comparing the prediction with the ground truth.

2. *Reflection*: The reflective agent is provided with the ground truth of future question correctness, and asked to predict some future answers' correctness.

3. *Testing*: They utilise the reflection, along with the past and future learning history to ask another novice agent which has not experienced the ground truth to make a new prediction.

4. *Iteration*: Finally, they obtain the prediction accuracy by comparing the predictions made by the novice agent with the ground truth. If the prediction accuracy is lower than the initial prediction accuracy, they ask the reflective agent to reflect in a different direction in the next iteration. Otherwise, they inform the agent that the accuracy has improved and that it could reflect towards the similar direction. The iteration stops when the novice agent achieves 100% accuracy or when they reach the maximum number of iterations.

After going through this process, they select the reflection with the highest prediction accuracy by the novice agent as the best reflection and log it into their reflection database. This database will then be used to augment existing prompting-based LLMs by giving example demonstrations or augmenting fine-tuning-based models to provide reflections.

Implementing the iterative reflection in TIR improves the simulation performance by adjusting the reasoning process of LLMs during reflection. This iterative reflection helps LLMs overcome the biases which occur due to the LLMs' pre-training corpus, and iteratively adjusts the reasoning process during reflection regarding the causality of how course stimuli modulate students' behaviours while also respecting the students' past history.

The authors evaluated the effectiveness of the TIR module by integrating it into both prompting-based and fine-tuning-based LLMs, and comparing their performance against the deep learning baselines. Specifically, they applied TIR to standard prompting and CoT prompting, and a fine-tuned Bidirectional Encoder Representations from Transformers (BERT)-based model (BertKT). These were benchmarked against five widely used deep learning knowledge tracing models: Attentive Knowledge Tracing (AKT) (Ghosh et al. (2020)), Simple Knowledge Tracing (SimpleKT) (Liu et al. (2023)), Adversarial Training Knowledge Tracing (ATKT) (Guo et al. (2021)), Deep Knowledge Tracing (DKT) (Piech et al. (2015)) and Dynamic Key-Value Memory Networks (DKVMN) (Zhang et al. (2017)).

Metric	GPT4o-Mini (Without/With TIR)			Deep Learning Models				
	Standard	CoT	BertKT	DKT	AKT	ATKT	DKVMN	SimpleKT
Accuracy	0.6025+0.0469	0.6222-0.0049	**0.6074+0.0938**	0.6351	0.6171	0.6396	0.6171	0.6772
F1 score	0.5128+0.1346	0.5610+0.0341	**0.6110+0.0770**	0.6352	0.6051	0.6390	0.6051	0.6698

Figure 3.1: Simulation results on EduAgent dataset (Source: Xu et al. (2025))

From their evaluation (as seen in Figure 3.1), they found that the integration of the TIR module achieved better simulation accuracy and F1 score then all of the deep learning baseline models. With the TIR module, the best performing model was the BertKnowledge Tracing (KT) model, whilst the best performing LLM-based model without the TIR module was CoT-based prompting.

In Hu et al. (2025)'s research paper, they attempt to tackle novice teacher's struggles to foresee students' needs and potential learning challenges. These struggles tend to lead to generic teaching plans, sometimes assembled from exemplary teaching plans by other teachers, and often lack depth and coherence. To solve this, they design prompt commands to make the LLM simulate classroom interactions between a teacher and students of varying ability levels. Then, using the LLM's reflection and error-correction capabilities, they generated teaching reflections based on the simulated teaching process. Afterward, they instruct the LLM to make corresponding improvements and optimizations to the original teaching plan based on the teaching process and reflection texts.

They split the process of the research design into three sections (as seen in Figure 3.2):

1. **Teaching Plan Dataset Construction**: For this phase, they categorise high school mathematics curriculum into four main knowledge modules: *Statistics*, *Functions*, *Algebra*, and *Geometry*. For each module, they select fifteen lessons. The teaching plans for those lessons were used as evaluation subjects. Using these teaching plans, they develop four datasets: (1) Dataset A contains teaching plans generated from prompt instructions based on Pedagogical Content Knowledge (PCK) theory and mathematical problem chains, (2) Dataset B contains teaching plans generated by GPT-4 using prompts without incorporating structured problem chains, (3) Dataset C consists of teaching plans sourced from experienced teachers with over ten years of teaching experience, and (4) Dataset G consists of teaching plans written by pre-service teachers in training.

2. **Teaching Plan Quality Enhancement**: For this phase, they simulate teacher-student classroom interactions, with varying student ability levels during and based on the content of the teaching plans in Dataset A. Then, they instruct GPT-4 to generate teaching reflections by integrating the original teaching plans with the simulated teaching process texts. Finally, they prompt GPT-4 to improve the original plans using the generated reflections. From this, they get another four datasets: (1) Dataset D which includes the improved plans taken from Dataset A, (2) Dataset E which contains further improved data from Dataset D (second round of improvement), (3) Dataset F consisting of improved teaching plans from Dataset A, but using Claude 3.5 instead of GPT-4, and (4) Dataset H which consists of enhanced teaching plans generated by GPT-4 from Dataset G. To improve clarity and instructional progression, they also categorise the problem chains designed by the LLM into three different formats: *context-based* which requires the LLM to create scenarios related and knowledge-introduction questions, *trap-based* which refers to a series of questions likely to cause students' mistakes, and *summary-based* which involves questions that guide students to review and summarise the knowledge learned.

3. **Teaching Plan Evaluation**: Finally for this phase, they conduct manual scoring and analysis of the eight previously mentioned teaching plan datasets. They designed an evaluation framework comprising of nine categories and nineteen dimensions. These categories include: **problem chains**, **teaching activities**, **content knowledge**, **teaching methods and strategies**, **teaching evaluation**, **interdisciplinarity**, **practical value**, **scope beyond the syllabus**, and **overall rating**. For manual assessment, they use a Likert 8-point scale.

To test the created teaching plans, they conducted a descriptive statistical analysis of sixteen evaluation dimensions on a total sample size of four hundred eighty teaching plans (two hundred forty before enhancement and after enhancement). From these tests, they detail a number of results. Firstly, the results gathered indicate that the directly generated teaching plans by GPT-4 in previous studies was less effective then the teaching plans generated using their method. They also showed that teaching plans written by pre-service teachers were improved significantly after being enhanced by the LLM, with the scores in the practical value dimension (Dimension V1) approaching the teaching plans improved through two rounds of enhancement. They also note areas in which the

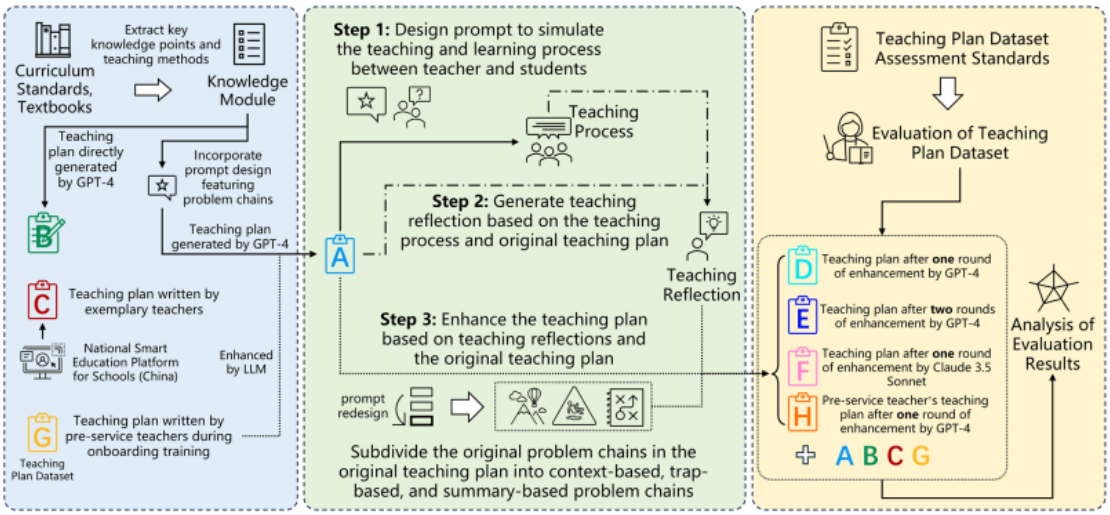


Figure 3.2: Construction and Evaluation of the Teaching Plan Dataset. (Source: Hu et al. (2025))

enhanced teaching plans fell short compared to the teaching plans written by human teachers, such as designing learning activities that promote teacher-student interactions (Dimension A1). For the disciplinary history and culture dimension (Dimension C3), they note that, despite outperforming humans in relative terms, the scores were still very low, signifying a clear room for improvement.

3.2 | LLMs in Education

Building on exploring AI's diverse roles within educational sectors, this section focuses on a specialised form of AI technology: LLMs in education. We will delve into several papers that successfully implement LLMs in many educational scenarios, highlighting their capabilities and versatility.

In Kaneko and Okazaki (2024)'s research, they attempt to tackle generating explanations for GEC in natural language through PI. For GEC systems to be valuable for learners, it is very essential for each corrected span to have a natural language explanation that is precisely aligned with it. To tackle this challenge, they insert prompts of edits into the generation process, which explicitly guides the LLM to generate explanations for all the corrections. The describe the process as follows:

1. The LLM receives the instruction to rewrite the input text into grammatically correct text and provide explanations for the corrections. For example, they inputted "*What*

is the difference between genetic disorder and other disorders.", and the LLM outputted "*What is the difference between genetic disorders and other disorders?*".

2. Then they compute the token alignment between the corrected output text generated by the LLM and the input text, allowing them to extract the edits. They also number the edits made to make it easier to distinguish each edit. Using the above example, they get (1. "*disorder* → *disorders*") and (2. ". → *?*").

3. Finally, they provide the extracted edits consecutively to the LLM, prompting it to generate an explanation for each edit.

Apart from PI, they also create the XGEC dataset which includes incorrect texts, correct texts and explanations for each edit. The XGEC dataset is built on data from:

- The NUCLE (Dahlmeier et al. (2013)) and CoNLL2013 (Ng et al. (2013)) datasets. These datasets contain one correct text per incorrect text. From these datasets, they randomly selected 362 correct texts and manually annotated explanations for them
- The CoNLL2014 (Ng et al. (2014)), which contains multiple correct texts per incorrect text (this is due to it commonly being used to evaluate GEC models). This dataset consists of *a* and *b* datasets which were created by different annotators. To reduce the amount of test instances with low annotator agreement on editing results, they chose edits that are regarded as appropriate by most humans. This data set includes an additional eight annotations which brings it to a total of 10 annotations. They annotated explanations only for the corrected spans that had a score of 7 or more out of ten in terms of agreement, which resulted to a total of 444 correct texts.

For the above datasets, they assigned two native English speakers to write an explanation for each edit. These annotators were provided with incorrect texts, correct texts, and the corresponding edits, with 10 example explanations being provided to serve as reference.

For their experiment, they evaluate how well GPT-3.5 (text-davinci-003) and ChatGPT (gpt-3.5-turbo-16k) can generate explanations for grammatical corrections using PI. The models were prompted using sixteen few-shot examples, and their outputs were compared across three configurations: generating explanations after corrections with PI

		XGECa			XGECb		
		Precision	Recall	F1	Precision	Recall	F1
ChatGPT	Post w/ PI	**83.2**	**85.5**	**84.3**	**83.9**	**84.5**	**84.2**
	Post w/o PI	62.1	79.6	70.0	62.6	78.2	69.6
	Pre w/o PI	60.9	75.2	68.1	61.1	74.4	67.7
GPT-3.5	Post w/ IP	**81.2**	**83.8**	**82.4**	**82.0**	**83.0**	**82.5**
	Post w/o IP	61.2	79.4	69.1	61.8	78.1	69.0
	Pre w/o IP	59.9	75.6	67.7	60.7	75.5	68.1

Figure 3.3: The BERTScore (Zhang et al. (2020)) of GPT-3.5 and ChatGPT in generating explanations with and without PI on the XGEC test datasets. (Source: Kaneko and Okazaki (2024))

		Validity	Coverage
ChatGPT	Post w/ PI	**1.5**	**2.0**
	Post w/o PI	1.2	1.4
	Pre w/o PI	1.1	0.9
GPT-3.5	Post w/ PI	**1.4**	**2.0**
	Post w/o PI	1.1	1.5
	Pre w/o PI	1.1	1.0

Figure 3.4: Human evaluations of GPT-3.5 and ChatGPT with and without PI on the XGEC test dataset. (Source: Kaneko and Okazaki (2024))

(**Post w/ PI**), after corrections without PI (**Post w/o PI**), and before corrections without PI (**Pre w/o PI**). The alignment of edits was performed using the ERRor ANnotation Toolkit (ERRANT) toolkit (Bryant et al. (2017); Felice et al. (2016)), and BERTScore (Zhang et al. (2020)) was used for automatic evaluation.

In the experiments conducted, the results demonstrated that both GPT-3.5 and ChatGPT produced significantly better explanations using PI. In terms of automatic metrics (seen in Figure 3.3), the **Post w/ PI** setup outperformed the baselines on both XGECa and XGECb datasets. Furthermore, human evaluations gave higher scores to **Post w/ PI** for explanation validity and coverage (as seen in Figure 3.4). This result showed that providing LLMs with explicit edit information allowed them to explain corrections more accurately and comprehensively.

In addition to the aforementioned results, they also tested how the generated explanations could improve model performance when used as few-shot examples for GEC tasks (as seen in Figure 3.5). When the models were prompted with examples containing explanations (human-written and generated with PI) the performance was comparable,

		CoNLL2014	W&I	JFLEG
ChatGPT	Pre Human	**55.2**	51.2	61.7
	Post Human	54.8	**51.5**	61.5
	Pre PI	54.9	51.7	61.5
	Post PI	54.7	49.7	**61.8**
	No explanation	52.3	40.1	55.3
GPT-3.5	Pre Human	54.0	**44.2**	**57.8**
	Post Human	**54.5**	44.0	57.3
	Pre PI	53.7	**44.2**	57.1
	Post PI	54.1	39.9	57.1
	No explanation	50.1	35.8	53.7

Figure 3.5: The GEC performance of GPT-3.5 and ChatGPT when using explanation text as examples for few-shot methods. (Source: Kaneko and Okazaki (2024))

and both outperformed the no-explanation baseline. This shows that the PI-generated explanations can be used for downstream learning tasks.

Another implementation of LLMs in education is Bannò et al. (2024)'s alternative process for spoken GEC. Typical approaches for implementing spoken GEC includes three modules: 1) an Automatic Speech Recognition (ASR) module which transcribes the spoken content, 2) a DD module which removes disfluencies, such as interruptions, repetitions and hesitations, and 3) a GEC system tuned to handle speech transcriptions, used for correcting grammatical errors. In this paper, they introduce an "end-to-end" approach, in which they use OpenAI's Whisper, a speech recognition model, with the purpose of performing end-to-end spoken GEC and DD.

To test their implementation, they implement two systems, one being the proposed end-to-end system, and the other one being a traditional cascaded spoken GEC. To implement the E2E system, they adapt the Whisper model by tuning it separately on three types of manual references: 1) original ASR transcriptions with disfluencies labelled (**dsf**, Whisper$_{dsf}$), 2) fluent transcriptions with hesitations, false starts, etc. removed (**flt**, Whisper$_{flt}$), and 3) grammatically corrected transcriptions (**gec**, Whisper$_{gec}$) (as seen on the right in Figure 3.6). They also explore two methods for adapting the Whisper model: FT, in which all of the model parameters are updated, and SPT, in which a small amount of continuous vectors are inserted into the decoder embedding space and tuned on the training set whilst keeping the original model parameters fixed.

For the Cascaded system, they implement the typical ASR, DD amd GEC modules (as seen on the right in Figure 3.6).

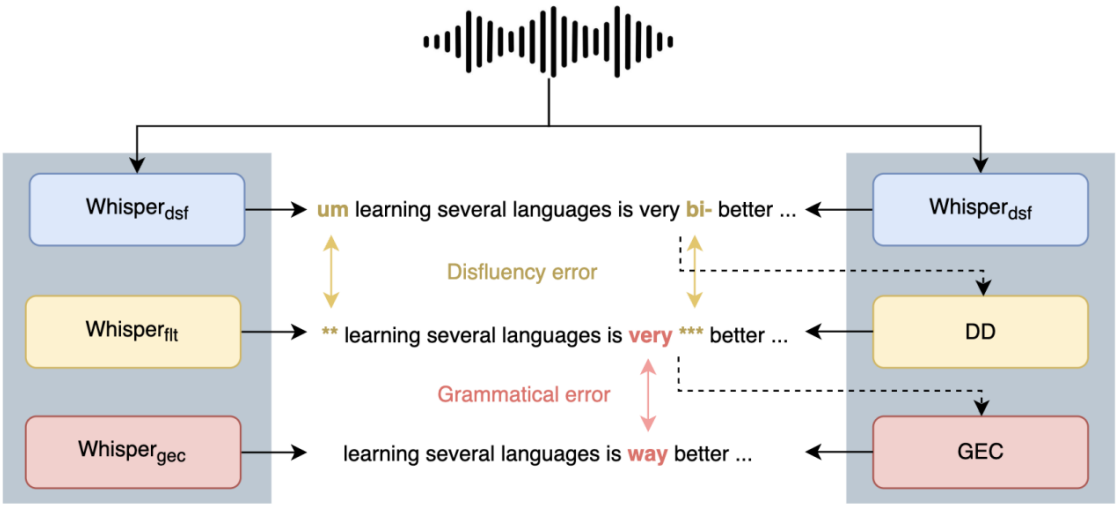


Figure 3.6: Illustration of an E2E SGEC system and a cascaded system. (Source: Bannò et al. (2024))

- **ASR**: They used the Whisper$_{dsf}$ model described above. Similarly, they train it on transcriptions with disfluencies labelled.
- **DD**: They employed a BERT-based token classifier for sequence tagging: $\mathbf{d}_{1:M} = BERT(w_{1:M})$ $p(r_{1:M}|w_{1:M}) = f_d(\mathbf{d}_m)$. For equations, they define $\mathbf{d}_m$ as the word embedding associated to word w_m; r_m as a binary tag which indicates whether word w_m is fluent or disfluent; f_d is trained to estimate how probably a given word w_m is fluent. Subsequently, they eliminated all words which identified as disfluencies from the transcriptions. The BERT-based model they utilised includes: a pre-trained BERT model (Sourced from the HuggingFace Transformer Library (Wolf et al. (2020)) under the name *bert-base-uncased*), a dropout layer, a dense layer with 768 nodes, an additional dropout layer, a second dense layer with 128 nodes, and the output layer.
- **GEC**: They perform GEC as a sequence-to-sequence task using the (Bidirectional and Auto-Regressive Transformers (BART) model (Lewis et al. (2019)), found from the HuggingFace Transformer library (Wolf et al. (2020)) as the GEC model.

To evaluate the implemented systems, they utilise several metrics:

- **ASR**: To evaluate the ASR system, they use WER which measures how many errors are in the transcription text produced by the ASR.

- **DD**: To evaluate the DD system, they utilise the WER as the primary metric, and Precision, Recall, and F_1 scores as additional metrics for feedback analysis. The additional metrics are not primary due to the fact that multiple ASR systems are being used with different decoded transcriptions.
- **GEC**: To evaluate the GEC system, they use both WER and Translation Edit Rate (TER) as primary evaluations metrics, while also employing Precision, Recall, and $F_{0.5}$ as additional metrics for feedback analysis. They chose these metrics for several reasons. The first reason they talk about is related to cascaded-style spoken language applications and the difficulty in comparing across systems when upstream modules are different. They also detail that evaluation metrics are not clearly defined for end-to-end trained spoken systems. Finally, they mention that transferring written-based metrics to spoken tasks is challenging due to the fact that end-to-end systems do not yield intermediate variables for assessment.

They detail three datasets which were used within the experiments:

- **Switchboard NXT**: A corpus consisting of manual transcriptions of 260 hours of telephone conversations by First Language (L1) American English speakers, including disfluency annotations and time-alignment information (Calhoun et al. (2010)). From this corpus, they removed any cases of unintelligible words, and filler words (such as "uh-hum"). In the experiment, they treat annotated disfluencies and hesitations/filler words as disfluencies.
- **Linguaskill**: Contains data obtained from candidate responses to the Speaking module of the Linguaskill tests for Second Language (L2) learners of English (Ludlow (2020)). The data is manually tagged with annotations about disfluencies and grammatical error corrections, and features 30 L1s and proficiency levels ranging from A2 to C of the Common European Framework of Reference (CEFR) (Council of Europe (2001)).
- **EFCAMDAT+BEA-2019**: EF-Cambridge Open Language Database (EFCAMDAT) is an L2 learner corpus comprising of 1,180,310 assignments written by 174,743 L2 learners (Geertzen et al. (2013)). The learner scripts in this corpus are annotated with proficiency scores, part-of-speech tags, and information on grammatical dependencies, and are partially corrected by human experts. They only used part of the responses due to the data containing noisy responses and incorrect annotations.

Model	dsf	flt
$\text{Whisper}_{\text{dsf}}$	10.62	14.83
$\text{Whisper}_{\text{dsf}}$+DD	-	10.86
$\text{Whisper}_{\text{flt}}$	13.83	**10.32**

Figure 3.7: Evaluation of Whisper FT performance against disfluent (dsf) and fluent (flt) manual references on Switchboard in terms of %WER. (Source: Bannò et al. (2024))

Building Educational Applications (BEA)-19 is a collection of written corpora tagged with GEC annotations. This was released by the organisers of a Workshop on Innovative Use of NLP for Building Educational Applications (Bryant et al. (2019)). For both datasets, punctuation and capitalisation have been removed to make them more similar to speech transcriptions.

Finally, they detail the setup of the models used for the experiments:

- **Whisper**: The pre-trained Whisper (small.en) model is tuned on the Linguaskill training set with different manual preferences for three tasks.
- **DD**: For the first part of the experiments, they train the model used on the Switchboard NXT data. For the other part, they train it on the re-annotated version of Switchboard. They also fine-tuned the model on Linguaskill data.
- **GEC**: The BART model used was trained on EFCAMDAT and BEA-2019 data. Then they fine-tuned it on the Linguaskill dataset.

From these experiments, they present a number of results. When evaluating on the Switchboard dataset, they saw that Whisper_{flt} for end-to-end ASR and DD outperformed the cascaded DD approach (as seen in Figure 3.7). They also evaluate the DD by calculating the deletions between the fluent ASR transcriptions from Whisper_{flt} and the disfluent transcriptions of Whisper_{dsf} (results in Figure 3.8). They also note that the discrepancy between the above results and WER performance is due to the fact that the cascaded system compares deletions based on single transcripts, whilst two different transcriptions produced with two different decoding processes are contrasted for the end-to-end system. Therefore, the disfluencies in the end-to-end system can be due to transcription differences and deletions in the fluent system from removing disfluencies.

For the Linguaskill dataset, they use the transcribed data to evaluate the end-to-end DD and GEC. From the results in Figure 3.9, they show that Whisper can be tuned to yield both end-to-end DD and GEC, with the best WER performance coming from the matched

DD Model	P	R	F_1
$\text{Whisper}_{\text{flt}} \xrightarrow{\texttt{del}} \text{Whisper}_{\text{dsf}}$	49.82	55.00	50.52
$\text{Whisper}_{\text{dsf}}\text{+DD} \xrightarrow{\texttt{del}} \text{Whisper}_{\text{dsf}}$	69.18	68.21	67.21

Figure 3.8: Evaluation of DD in terms of Precision, Recall, and F_1 scores on the Switchboard test set based on deletions. (Source: Bannò et al. (2024))

Model	dsf		flt		gec	
	FT	SPT	FT	SPT	FT	SPT
$\text{Whisper}_{\text{dsf}}$	**5.92**	6.36	9.97	10.58	19.17	19.58
$\text{Whisper}_{\text{dsf}}\text{+DD}$	—	—	6.31	6.82	—	—
$\text{Whisper}_{\text{flt}}$	9.22	9.55	**5.77**	6.34	14.89	15.29
$\text{Whisper}_{\text{gec}}$	13.73	12.51	10.37	9.26	**13.49**	14.22

Figure 3.9: Evaluation of Whisper FT and SPT performance against different menual references on Linguaskill in terms of %WER. (Source: Bannò et al. (2024))

System	ASR Model	gec	
		WER	TER
Baseline	$\text{Whisper}_{\text{dsf}}$	19.17	18.74
	$\text{Whisper}_{\text{flt}}$	14.89	14.49
Cascaded	$\text{Whisper}_{\text{dsf}}\text{+DD+GEC}$	13.34	12.96
	$\text{Whisper}_{\text{flt}}\text{+GEC}$	**12.96**	**12.54**
E2E	$\text{Whisper}_{\text{gec}}$	13.49	13.08

Figure 3.10: GEC results with FT baseline, cascaded, and E2E systems. (Source: Bannò et al. (2024))

system. For the remainder of the experiments, they focus on FT due to it constantly outperforming SPT. In Figure 3.10, they note that the best-performing model is the cascaded system utilising the Whisper_{flt} model (Whisper_{flt} + GEC). They note that this result must be taken into context that the GEC module used in the cascaded systems is trained on a large amount of GEC data, and fine-tuned on Linguaskill transcriptions, whilst the Whisper_{gec} was trained only on the Linguaskill data. That being said, Whisper_{gec} still achieves comparable performance to the cascaded system. Looking into Figure 3.11, they note that Whisper_{flt} is mainly performing deletions when compared to manual disfluent transcriptions, and Whisper_{gec} mostly performs substitutions, and to a lesser degree, insertions and deletions.

Finally, they discuss the analysis on the feedback given to the learner of where and how a learner has been disfluent or made grammatical errors. When evaluating DD on the Linguaskill data in terms of Precision, Recall and F_1 score based on deletions between Whisper_{flt} and Whisper_{dsf} transcripts, they noticed a discrepancy between the performance reported in Figure 3.12 and the WER performance in Figure 3.9. They

Model	WER (Sub/Del/Ins)		
	dsf	flt	gec
$\text{Whisper}_{\text{dsf}}$	3.3/1.3/1.4	3.2/0.8/5.9	8.2/3.2/7.7
$\text{Whisper}_{\text{flt}}$	2.9/5.4/0.9	3.0/1.3/1.5	7.8/3.8/3.4
$\text{Whisper}_{\text{gec}}$	5.3/6.2/2.3	5.4/2.1/2.9	6.8/3.2/3.5

Figure 3.11: WER breakdown (Sub/Del/Ins) of Whisper transcription against different manual references. (Source: Bannò et al. (2024))

DD Model	Strategy	P	R	F_1
$\text{Whisper}_{\text{flt}} \xrightarrow{\texttt{del}} \text{Whisper}_{\text{dsf}}$	SPT	66.63	70.97	66.84
	FT	61.02	68.11	62.30
$\text{Whisper}_{\text{dsf}}\text{+DD} \xrightarrow{\texttt{del}} \text{Whisper}_{\text{dsf}}$	SPT	74.95	75.00	73.28
	FT	74.94	75.05	73.35

Figure 3.12: Evaluation of DD in terms of Precision, Recall, and F_1 scores on the Linguaskill test set based on deletions. (Source: Bannò et al. (2024))

GEC Model	Strategy	P	R	$F_{0.5}$
$\text{Whisper}_{\text{gec}} \xrightarrow{\texttt{gec}} \text{Whisper}_{\text{flt}}$	SPT	25.41	14.60	22.13
	FT	27.77	22.31	26.40
$\text{Whisper}_{\text{flt}}\text{+GEC} \xrightarrow{\texttt{gec}} \text{Whisper}_{\text{flt}}$	SPT	43.53	25.91	38.31
	FT	44.70	27.53	39.74
$\text{Manual}_{\text{flt}}\text{+GEC} \xrightarrow{\texttt{gec}} \text{Manual}_{\text{flt}}$	-	58.64	35.84	52.02

Figure 3.13: Evaluation of GEC in terms of Precision, Recall, and $F_{0.5}$ scores on the Linguaskill test set based on GEC edits. (Source: Bannò et al. (2024))

evaluate the GEC modules' performance in terms of Precision, Recall, and $F_{0.5}$ scores by considering GEC edits between the Whisper_{flt} transcripts Whisper_{gec} one. They extract these using the ERRANT (Bryant et al. (2017)). They note that results observed from the GEC models (as seen in Figure 3.13) are in line with their performance in terms of WER and TER shown in Figure 3.10. They also show the results of the GEC evaluation for Precision, Recall and $F_{0.5}$ scores considering all the 9 most common ERRANT edit labels individually for the end-to-end and cascaded GEC systems (as seen in Figure 3.14). From this, they note that when looking at `R:VERB` and `R:NOUN`, it makes it evident that more training data is needed for the end-to-end system. Those The labels mentioned indicate errors related to word usage and require the word spoken to be replaced with a different word. On the other hand, the results on other edit labels fall more in line with the cascaded system, such as `R:VERB:FORM`. They suggest that this type of switch may be observed in the Whisper training data.

To conclude, the authors detail that although the cascaded system performed the best, their proposed end-to-end system achieves comparable performance, which is particularly

Error	$W_{gec} \xrightarrow{gec} W_{flt}$			W_{flt}+GEC $\xrightarrow{gec} W_{flt}$		
	P	R	$F_{0.5}$	P	R	$F_{0.5}$
R:PREP	54.45	31.81	47.66	54.77	40.54	51.18
M:DET	39.69	40.96	39.94	54.70	43.93	52.14
R:NOUN:NUM	46.96	40.18	45.43	61.20	43.79	56.69
R:VERB:TENSE	35.39	19.87	30.61	45.96	23.34	38.50
R:VERB	10.74	4.70	8.55	42.06	15.59	31.40
U:DET	41.33	34.78	39.83	45.21	46.89	45.54
R:DET	22.58	18.26	21.56	41.54	23.48	36.00
R:NOUN	5.53	5.69	5.56	36.21	9.95	23.70
R:VERB:FORM	44.25	40.10	43.35	50.29	44.79	49.06

Figure 3.14: Evaluation of GEC in terms of Precision, Recall, and $F_{0.5}$ scores on the Linguaskill test set focusing on individual GEC edits. (Source: Bannò et al. (2024))

interesting considering the ASR model was fine-tuned on a limited quantity of data. They also note that one of the challenges for the end-to-end systems is feedback for learners, in which the end-to-end SGEC systems yield only grammatically correct speech, as opposed to cascaded systems which are able to yield grammar edits. That being said, they do note that edits can be obtained by comparing fluent speech versus grammatically correct speech in the two end-to-end systems, but this limits the quality of the feedback. For future work, their intention is to explore other alternative foundational models, as well as conduct more extensive analysis of feedback options.

The study by Tran et al. (2024) introduces Readability-Controlled Instruction Learning (ReadCtrl), a method to instruction-tune LLMs to tailor to users' readability levels. This dynamic framework enables LLMs to generate content at various (near continuous level) complexity levels, enhancing their versatility across different applications.

They design their methodology to evaluate the effectiveness of instruction tuning across three tasks; **Text Simplification**, **Paraphrase Generation**, and **Semantic Entailment Generation**. These tasks were chosen on purpose to test the model's capability in adjusting the complexity of its output to match specified readability levels.

- **Text Simplification**: The aim of this task is to reduce the readability level of the given input text, making it more accessible to a wider audience with varying comprehension skills. This task challenges the model to simplify complex text whilst preserving its essential content and meaning.
- **Paraphrase Generation**: In this task, the model is tasked with rewording the given text to produce a paraphrase that maintains the readability of the original text. This requires a nuanced understanding of language to ensure the output remains true to the input's complexity and style.

- **Semantic Entailment Generation**: For this task, the model creates text that semantically follows from the given input, with the flexibility to increase or decrease the readability level. The model must grasp underlying meaning of the input text and generate output that logically entails the input.

They employ the instruction tuning to help the model control the readability for all the aforementioned tasks. This method provides explicit instructions to the model to control the output text's readability score, ensuring that generated content aligns with the intended complexity level for the target audience.

To achieve the desired readability level across the above tasks, they utilise a concise instruction which forces the model to generate content that semantically follows from the given input and aligns with a specified readability level. The instruction they give is as follows:

"Given an input text, please output an entailment with a readability score around target readability score." - Tran et al. (2024)

They evaluate the readability of the generated text using a variety of established readability metrics, calculating the following readability scores:

- **Gunning Fog Index**: Estimates the years of formal education required to understand the text on first reading (Gunning (1952))
- **Flesch-Kincaid Grade Level**: Translates the US grade level needed to comprehend the text (Kincaid et al. (1975))
- **Automated Readability Index**: Outputs a score correlating to the US grade level necessary for understanding (Smith and Senter (1967)).
- **Coleman-Liau Index**: Estimates the US grade level needed to comprehend the text using letter count instead of syllable count (Coleman and Liau (1975)).

The metrics mentioned above were chosen due to their varying approaches to assessing text complexity. Using these scores, they derive an average Reading Grade Level (RGL) to represent the text's overall readability. Then, by adjusting the instruction based on the target RGL, they are able to fine-tune the complexity of the output, making their approach adaptable to a wide range of applications.

To evaluate their proposed framework, they utilise six distinct datasets, two for each task to explore the model's capabilities in both seen (models tested using datasets which they were trained on) and unseen (models tested against new datasets which they weren't trained on) settings.

- **Text Simplification**: They utilise the ASSET (Alva-Manchego et al. (2020)) dataset, a diverse corpus for automatic sentence simplification ideal for instruction tuning and evaluation in seen settings, and the WikiSmall (Zhu et al. (2010)) dataset, which offers a different collection of simplified sentences derived from Wikipedia articles, used for evaluating performance in an unseen setting.
- **Paraphrase Generation**: They utilise the Paraphrase Adversaries from Word Scrambling (PAWS) (Zhang et al. (2019)) dataset which contains pairs of sentences paraphrasing each other, suitable for training and seen setting evaluations, and the Microsoft Research Paraphrase Corpus (MRPC) (Dolan and Brockett (2005)) dataset, which offers a collection of sentence pairs sourced from online news sources labelled as paraphrases or not, useful to test the model in unseen settings.
- **Semantic Entailment Generation**: For this task, they use the Stanford Natural Language Inference (SNLI) (Bowman et al. (2015)) dataset, containing a large collection of sentence pairs annotated with textual entailment information, used for instruction tuning and seen setting evaluation, and the MultiGenre natural Language Inference (MultiNLI) (Williams et al. (2018)) dataset which extends SNLI to a broader range of genres and contexts, useful for unseen settings.

They also employ various multifaceted metrics which assess a number of different aspects of the generated texts, this includes:

- **Average Readability Score**: Calculates the average readability level of generated texts.
- **Readability Gap (Delta)**: Measures the difference between the requested readability level and the actual readability level of the generated text.
- **Factuality**: Assess the factual alignment of the generated text with the source content or input (based on the SummaC methodology by Laban et al. (2022)).

- **Consistency and Coherence**: Measured using criteria from the UniEval (Zhong et al. (2022)) framework. It provides standardised methods for evaluating logical consistency and coherence of a piece of text. This ensures that the text is both logically structured and coherent.
- **SARI**: The System output Against References and the Input sentence (SARI) (Xu et al. (2016)) metric measures the model's ability to produce simplified text which is both accurate and helpful by comparing the generated output against both the original text and reference simplifications.
- **BLEU**: The BLEU (Papineni et al. (2002)) metric quantifies the linguistic similarity between the generated texts and reference texts. This indicates the model's capability to produce coherent and contextually appropriate content.

They specify the models used to understand the efficacy in handling the aforementioned tasks. These models include (1) **GPT-3.5** , (2) **GPT-4**, (3) **Claude-3**, and (4) **Mistral 7B ReadCtrl** model. The Mistral 7B ReadCtrl model is their own proposed model which has been instruction-tuned to adjust the readability level of generated texts.

From the above experiments, they present several results. In Figure 3.15, they present a comparison of performance between the four aforementioned models on all tasks. They first look at the performance in seen tasks which involves instruction tuning using ASSET, SNLI and PAWS. Their proposed Mistral-ReadCtrl model outperforms the other models in the **Readability Gap** and **Factuality** scores, demonstrating the model's capability of adhering to target readability levels and factual consistency. For **Consistency** and **Coherence**, which measure the logical flow and structural soundness of texts, the Mistral-ReadCtrl model performed better then most models, but was outperformed by the GPT-4 model in the PAWS dataset. Finally, for the **BLEU** and **SARI** metrics, important for evaluating the linguistic and contextual appropriateness in text simplification and paraphrase generation, Mistral-ReadCtrl scored the highest.

For unseen tasks, they look at the performance of the models using the WikiSmall, MultiNLI, and MRPC datasets. Similar to the results in the seen tasks, the Mistral-ReadCtrl managed to achieved the lowest **Readability Gap** and the highest **Factuality**, **Coherence** and **BLEU** scores throughout the tasks. For **SARI**, the Mistral-ReadCtrl model slightly trails behind the GPT-4 model on the MultiNLI dataset. From these

Models	Readability Gap↓	Factuality↑	Consistency↑	Coherence↑	BLEU↑	SARI↑
ASSET (seen) | WikiSmall (unseen) - Text Simplification						
Claude-3	3.6323 | 4.53	0.5221 | 0.4612	0.9301 | 0.9391	0.934 | 0.9396	0.1874 | 0.1606	40.6964 | 32.9996
GPT-3.5	2.8635 | 3.12	0.7231 | 0.6721	0.9641 | 0.9401	0.9648 | 0.9231	0.2739 | 0.194	41.0061 | 33.9842
GPT-4	2.7465 | 2.69	0.6547 | 0.5892	**0.9688** | **0.9556**	**0.9687** | 0.949	0.2061 | 0.1666	39.7319 | 31.4657
Mistral-ReadCtrl	**1.8384** | **2.09**	**0.7687** | **0.7168**	0.9423 | 0.9477	0.9653 | **0.9763**	**0.4317** | **0.4321**	**49.3521** | **42.1033**
SNLI (seen) | MultiNLI (unseen) - Semantic Entailment Generation						
Claude-3	4.6433 | 5.64	0.5102 | 0.3904	0.919 | 0.8292	0.9331 | 0.8346	0.0446 | 0.0303	48.3281 | 44.4344
GPT-3.5	2.8333 | 6.7	0.5176 | 0.3967	0.9049 | 0.8829	0.8982 | 0.896	0.0875 | 0.0378	51.0201 | 44.0607
GPT-4	2.4733 | 3.36	0.5632 | 0.5167	0.9488 | **0.8961**	0.9382 | 0.8879	0.105 | 0.0562	**52.1153** | **46.4204**
Mistral-ReadCtrl	**1.8733** | **2.21**	**0.7406** | **0.6542**	**0.9491** | 0.8804	**0.9437** | **0.9122**	**0.183** | **0.1137**	51.6644 | 43.8289
PAWS (seen) | MRPC (unseen) - Paraphrase Generation						
Claude-3	2.4333 | 2.61	0.5141 | 0.4736	0.921 | 0.9154	0.9183 | 0.9012	0.2393 | 0.1679	38.3459 | 36.7783
GPT-3.5	1.5433 | 2.64	0.7443 | 0.5868	0.9761 | **0.9683**	0.9746 | 0.9679	0.3873 | 0.2059	37.9808 | 37.3417
GPT-4	1.4467 | 2.19	0.7085 | 0.5203	**0.979** | 0.9635	**0.978** | 0.9639	0.3122 | 0.153	34.3525 | 34.8477
Mistral-ReadCtrl	**0.6367** | **1.66**	**0.7871** | **0.8184**	0.9677 | 0.9669	0.9735 | **0.9769**	**0.6649** | **0.3798**	**60.5332** | **44.4327**

Figure 3.15: Main results for seen | unseen tasks in ReadCtrl. (Source: Tran et al. (2024))

results, they confirm Mistral-ReadCtrl's generalisation capabilities across the various tasks.

Apart from the calculation of the above metrics, they also conduct human evaluation using five human evaluators and one expert evaluator. For this, they give the human evaluators thirty six pieces of data, built by randomly sampling six data from the test split of each dataset detailed above. They give the annotators the following detailed instructions:

"You are evaluating two systems, both of which are trying to convert inputs to specific readability requirements to produce output suitable for the user. I will show you the input and output of the two systems on grade 2/5/8/11, respectively. Tell me which system's output you prefer by specifying system 1 or system 2 or tie if the quality is the same. Please explain the reason for your preference." - Tran et al. (2024)

Each time, they randomly shuffle the outputs of two systems (Mistral-ReadCtrl and GPT-4), and the annotators can choose the output which better meets the readability requirements and has higher output quality. They can also choose both if they think the outputs of the two systems are tied. After they get the judgments from multiple people per instance, they calculate the win rate of the models by counting them individually. After the above preference evaluation, they then worked with one Linguistics expert to annotate readability control strategies. They summarised four different readability control strategies for each grade level (as seen in Figure 3.16), and asked the expert to select the strategies which appeared for the outputs of the Mistral 7B ReadCtrl system.

Grade 2	
Employ short, straightforward sentence structures	100%
Focus only on essential details, omitting unnecessary complexity	85.7%
Use very simple vocabulary and avoid complex words	76.2%
Break down information into clear sequential steps	35.7%
Grade 5	
Introduce some more varied and content-specific vocabulary	71.4%
Use longer sentences with conjunctions to combine ideas	57.1%
Provide additional context and relevant details	28.6%
Explain concepts more directly instead of narratives	23.8%
Grade 8	
Use complex sentence structures like passive voice	66.7%
Employ richer descriptive language and vivid details	54.8%
Incorporate academic and technical terminology	47.6%
Establish clear logical connections between ideas	21.4%
Grade 11	
Construct elaborate compound-complex sentences	42.9%
Use sophisticated vocabulary from all domains	40.5%
Write with consistent formality and academic tone	33.3%
Employ advanced stylistic techniques like figurative language	23.8%

Figure 3.16: Readability control strategies for Mistral 7B ReadCtrl. The number represents what proportion of the system output in the corresponding grade level uses the corresponding method to adjust the readability. (Source: Tran et al. (2024))

From the results of the Human Evaluation (as seen in Figure 3.17), they note that human evaluators preferred the Mistral-ReadCtrl model over the GPT-4 model, with an overall win rate of 49.4%. When delving into the human annotations, they shed light on the operational tactics of Mistral-ReadCtrl. For example, when the model caters to the Grade 2 readability, it used short, straightforward sentence structures 100% of the time, focused on essential details 85.7% of the time, used very simple vocabulary and avoided complex words 76.2% of the time, and broke down the information in the sentences into clear sequential steps in 35.7% of the instances. For Grade 5 and 8, the model introduced content specific vocabulary (Grade 5) in 71.4% of the outputs and complex sentence structures (Grade 8) in 66.7% of the outputs, which they note illustrates the model's dexterity in scaling complexity according to the readability demands.

To conclude, their proposed approached enhances the readability of LLMs by adjusting content complexity dynamically to suit a variety of readability requirements. By outperforming the models used in the experiments, Mistral-ReadCtrl demonstrated its capability to produce nuanced, high-quality outputs, showing promise of personalized content generation. They mention a number of limitations, such as a resource limitation which limited their analysis to Text Simplification, Paraphrase Generation and Semantic Entailment Generation. They note that due to the resource constraints, they were also unable to have actual grade 2, 5, 8 and 11 students provide pairwise preference feedback during the human evaluation.

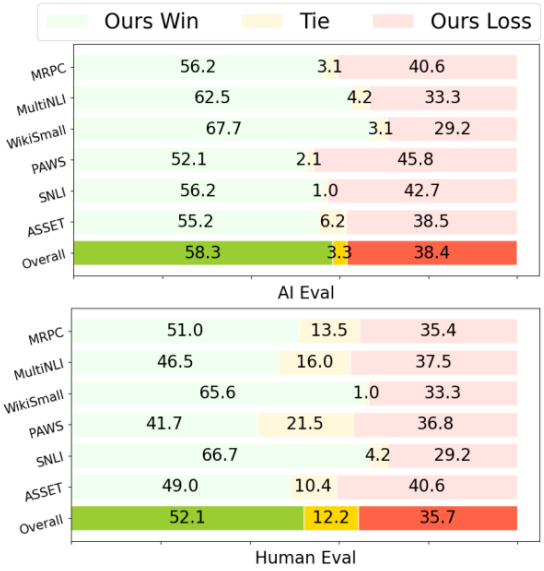


Figure 3.17: Win rate (%) for Mistral-ReadCtrl vs GPT-4 (3 shots) using AI (Claude3 and GPT-3.5) and Human evaluation. (Source: Tran et al. (2024))

3.3 | Intelligent Agents in Education

PCAs refer to conversational agents which act as various roles (instructors, peers, motivators) with whom learners can talk to using natural language. PCAs can be used in diverse subjects, grades and pedagogies. With LLMs, teachers can be empowered to build PCAs customised for their students. Due to these students having different prior knowledge and motivation levels, these PCAs have to be reviewed and adapted to fit the variety of students. To tackle this, Jin et al. (2025) present TeachTune, a web-based tool in which teachers can create simulated students and review PCAs by observing automated chats between the PCAs and simulated students.

Teachers can build these PCAs with a graph-like state machine representation (as seen in Figure 3.18). The PCA's state machine starts off with a root node that consists of the PCA's start message to the students, and the instruction it initially follows. The PCA itself is an LLM-based agent which is prompted conditionally with the state machine. The state of the PCA is determined by a master LLM agent. This master agent monitors the conversation happening between the PCA and a student, and decides whether the PCA's state should remain in the same node, or if it should transition to one of its child nodes.

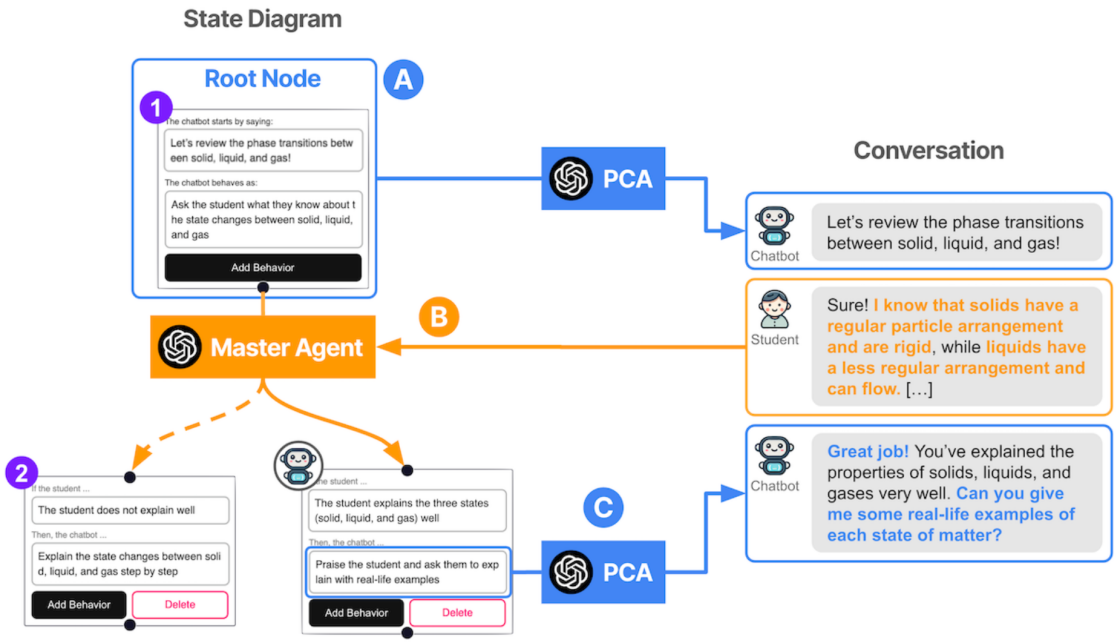


Figure 3.18: A PCA follows the dialogue flow defined in its state diagram. Nodes represent the PCA's utterance, and edges represent the potential response path of simulated students. The root node (A) contains the PCA's starting message and initial behaviour. Based on a student's response, the master agent keeps the current state or changes the active node to one of the connected nodes (B). The next active node determines the PCA's subsequent response (C). (Source: Jin et al. (2025))

To review the robustness of their PCAs, they allow teachers to test using three different methods: direct chat, single-turn test cases, and automated chat. They discuss how they implemented the last method, in which teachers can review PCAs in two steps:

1. **Templated Student Profile Creation**: In this step, teachers have to define what types of students they review against. When creating each student profile, teachers are allowed to specify the student's initial knowledge (through check-marking knowledge components provided) and configure the student's personality (through rating the trait inventories on a five-point Likert scale). With this information, TeachTune generates a natural language description, known as the **trait overview** of the student, which teachers can edit to correct or add more context about the student. Once the teachers have created various student profiles to review against, they can leverage it over the iterative PCA design process.

2. **Automated Chat**: Teachers then get to select one of the student profiles they created to generate a lesson conversation between the simulated student and their PCAs. PCAs start conversations, and the state marker on the state diagram transits

in real-time throughout the conversation. TeachTune generates six messages (i.e., three turns) between the PCAs and simulated students at a time, and teachers can keep generating further conversation by clicking the "Generate Conversation" button. When teachers find any corner cases which the designed PCA didn't cover, they can then add a node that describes the case and appropriate instruction for the PCAs.

They also propose a *Personalised Reflect-Respond* (as seen in Figure 3.19) LLM pipeline used to simulate conversations with specific student profiles, extended from the *Reflect-Respond* pipeline detailed in Jin et al. (2024). The *Reflect-Respond* pipeline generates a simulated student's response in two steps: *Reflect* updates the knowledge state by activating relevant components, while *Respond* produces a likely reply based on the updated state and conversation history. Jin et al. (2025) add onto this pipeline by giving an LLM additional instruction in the *Respond* step. Before the runtime of the *Reflect-Respond* steps, they add the *Interpret* step which translates trait scores into a **trait overview** containing a comprehensive summary and reasoning of how the student should behave. Once the overview is edited and confirmed to be correct by the teacher, it is then passed to the *Respond* step so that the LLM takes the student traits into account, and the additions made to the conversational context and knowledge state.

Their evaluation of the aforementioned implementations was separated into two: (1) in the **Technical Evaluation**, they created nine simulated students of diversely sampled knowledge and trait configurations, and asked ten teachers to predict their configuration through direct chats and pre-generated conversations, and (2) in the **User Study**, they ran a between-subjects user study with thirty teachers and observed how the student profile template and simulated students helped the design and reviewing of PCAs. From these experiments, they notice the following results:

1. **Technical Evaluation**: The results of this evaluation showed that the *Personalised Reflect-Respond* can instruct an LLM to simulate a students' behaviour given a specific knowledge state and traits precisely. More specifically, they noted that: (1) The knowledge bias was small, meaning that the evaluators correctly identified the knowledge components most of the time, except for two specific cases; (2) The gap between the configured and perceived levels of student traits was also small; (3) The evaluators reported that the simulated students behaved as naturally as real students,

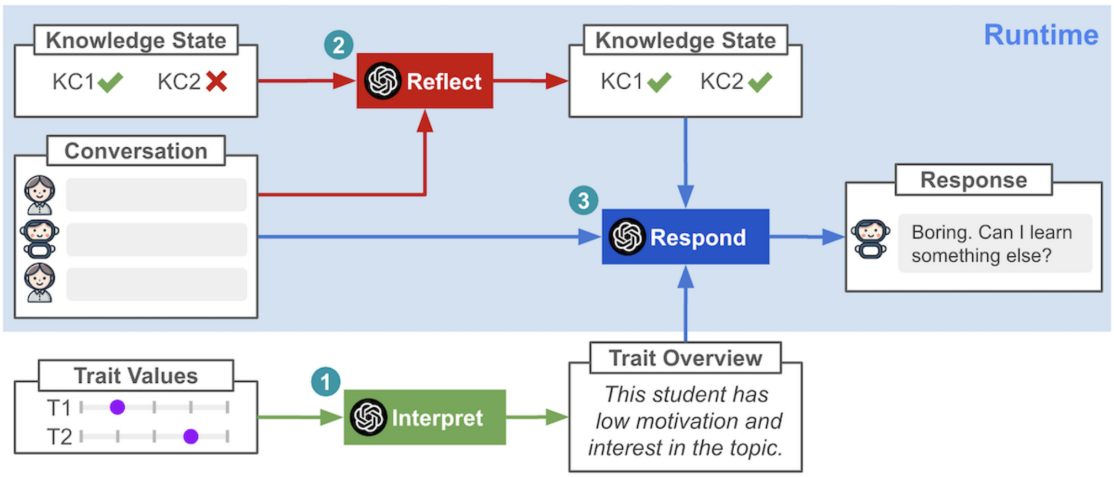


Figure 3.19: The *Personalized Reflect-Respond* pipeline. The pipeline interprets the student's trait values and creates a trait overview (1), and the previous conversation history is used to update the knowledge state through the reflect pipeline (2). Afterward, the Respond pipeline takes the conversation, updated knowledge state, and the trait overview to generate the response (3). The blue background is a runtime area where the components inside change throughout a conversation. The trait overview is created once before the runtime. (Source: Jin et al. (2025))

and are helpful for teaching training; (4) The *Interpret* step within the *Personalised Reflect-Respond* pipeline helped with generating more believable behaviours.

2. **User Study**: They detail several results from this evaluation, including: (1) Autochat alleviated teachers' burden in ideation and repeated tasks; (2) Participants who used Autochat considered more unique student profiles; (3) Direct chats, test cases, and automated chats complemented each other, with each type of chat helping in different ways, such as direct chats helping in reviewing specific scenarios, automated chats being used for later exploration stages and coverage tests, and test cases being helpful for node-oriented debugging of PCAs; (4) The difference in PCA qualities among conditions was insignificant, which they suggest could mean that Autochat participants required additional support to incorporate their insights and findings from automated chats into their PCA design.

Apart from the promising results, they also outline several limitations with their implementation. First, they mention that due to the study focusing on evaluating the quality of PCAs with experts, this study did not confirm the pedagogical effect of PCAs on students' learning gain and attitude. They detail that student-involved studies could help reveal the gap between the teachers' expectations and the students' actual learning. Second, they mention that their technical evaluation and user study were limited to a single subject and

learning topic (phase transitions in science). Even though they expect their findings will generalise to other STEM fields with well-defined knowledge components, they suspect that other humanities subjects may require additional support (example, simulating students' cultural backgrounds in literature classes). Lastly, they simulated a limited number of student traits only. Although their *Personalised Reflect-Respond* pipeline introduced a multifaceted student simulation that involves both knowledge and student traits, they acknowledge that more personal attributes of students need to be implemented for authentic simulated students. Moreover, they make the assumption that student traits are static throughout conversations, but in reality, actual students may change their attitudes with proper guidance, meaning that student traits should be as malleable as the knowledge state.

Student simulation in online education is important to address the dynamic learning behaviours of students with diverse backgrounds. Existing simulation models based on deep learning require a large amount of training date, lacking prior knowledge in educational contexts. LLMs may contain the prior knowledge since they are pre-trained form a large corpus, but directly prompting them is not robust, nor accurate enough to replicate the dynamic and multifaceted nature of student behaviours. To tackle this problem, Xu et al. (2024) present EduAgent, a novel generative agent framework which incorporates cognitive prior knowledge to guide the LLMs, and an annotated fine-grained large-scale dataset.

They first elaborate on the **EduAgent310** and **EduAgent705** datasets:

- **EduAgent310**: This dataset is an augmented version of Xu et al. (2023)'s dataset containing raw behaviour data of 310 students. Each student participated by watching a slide-based online lecture and answering post-lecture test questions.
- **EduAgent705**: This dataset was created to increase the original dataset's size and diversity, tackling the issue of collecting large-scale real behavioural data. EduAgent705 consists of 705 simulated students, each simulated using their proposed EduAgent framework. For each virtual student, they assign a diverse persona, modelled after the cognitive science findings that link personal characteristics to learning outcomes.

For the EduAgent, they detail the following simulation pipeline (as seen in Figure 3.20):

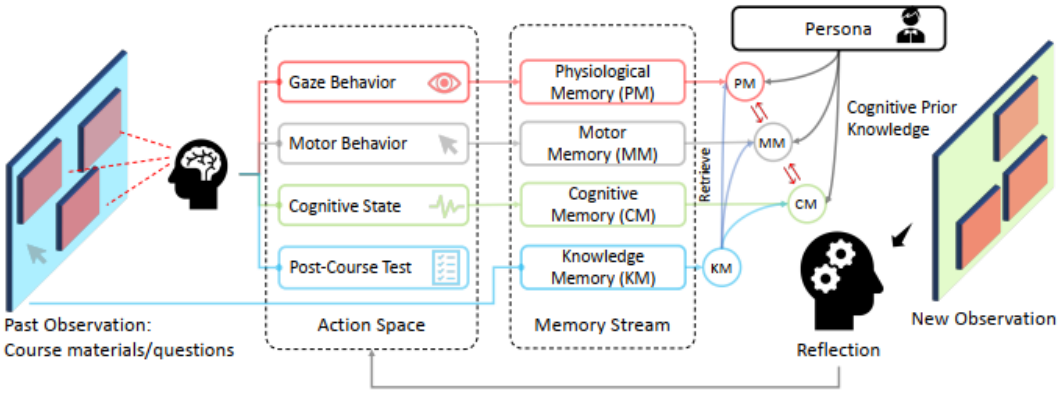


Figure 3.20: Our EduAgent framework. (Source: Xu et al. (2024))

1. First, they instantiate each student agent with one persona profile (as mentioned in the EduAgent705 dataset).

2. Then, they simulate the learning process for the agent slide by slide, following the order of the lecture. Each slide represents a simulation step, and within each step, the agent processes all transcript sentences (one per transcript)

3. For each transcript on a slide, the agent receives it as an input and outputs a trajectory of simulated learning behaviours, including gaze behaviours, motor control behaviours, cognitive status and answers to test questions related to the slide.

4. They then simulate the agent's learning process from the first to the last slide of the lecture in order, with each slide being a simulation step. For every slide, the agent receives its transcripts and outputs a trajectory of simulated learning behaviours (actions) for each transcript (each transcript is one sentence).

5. Before generating new actions, the agent reflects on the correlation among its persona, past actions, and past course materials. These reflections are guided using cognitive priors (Bourgin et al. (2019)).

6. The outcomes of the agent's reflection are then used to inform its actions on new course materials or test questions. Cognitive priors are embedded in the prompts to the LLM to guide its reasoning about (1) how and why behaviours in each memory layer are modulated by persona and course content, and (2) how different behaviours affect each other, preventing simple copying of past demonstrations.

Within the above pipeline, they also detail extra features within the simulation which they implement:

- **Gaze/Motor Behaviour Simulation**: The agent maps gaze and motor behaviours to specific Areas of Interest (AOIs) on each slide. The LLM is then prompted to output the AOI ID for each action, correlating the semantic confirmation from the transcript to the sensory behaviours. The agent reasons how these factors module gaze and motor actions based on memory and cognitive priors, then it simulates the behaviours for the current input.
- **Cognitive States Simulation**: For each transcript, the agent generates a numeric value (0 to 1) for each cognitive state factor. The agent reasons how persona, course content, and past gaze/motor behaviours modulate cognitive states, applying these insights to simulate current cognitive states.
- **Learning Performance Simulation**: For each post-lecture question, the agent selects one answer from four options, aiming to mimic individual student answering behaviour. The agent reasons how the answer correctness is affected by persona, cognitive states, and gaze/motor behaviours. If the agent predicts the student would answer correctly, it selects the correct answer; else, it estimates the most plausible incorrect choice based on behaviour patterns.

To evaluate the above implementations, they create two experiments:

- **Personalised Student Behaviour Prediction**: They evaluate the EduAgent's ability to predict the future student learning behaviours and outcomes, using the students' past learning behaviour as few-shot demonstration. They test this using several foundational models, including GPT-3.5, GPT-4, Gemini Pro Gem and Llama 2 70B lla. The results gathered from this experiment showed that the cognitive priors improved behaviour simulation, the specific memory components were important and removing any specific one degraded the performance. The GPT-4 model outperformed the other LLMs in accurately modelling behaviour correlations.
- **Virtual Generative Student Simulation**: They test whether the EduAgent exhibits reasonable learning behavioural patterns without real demonstration data. They also analyse the realism of behaviours using Pearson correlation between the

personas and the behavioural metrics. From this experiment, they note several results: (1) high-GPA student agents have better gaze/motor following and better test scores, (2) Cognitive states like curiosity and engagement aligned correctly with the personas, and (3) gaze/motor behaviours correlated well with cognitive states and performance.

They conclude their research by discussing a number of limitations with their implementation. The first limitation they mention is the fact that Student behavioural simulation is challenging due to the inherent noise and unpredictability of human behaviour. They also note that unlike simple persona mimicry or one-off behaviour limitation, EduAgent should jointly simulate personas, behaviours and course content. Finally, they note that some generated behaviours do not align with real-world cognitive science findings, which they attribute to limitations in how much the LLM can reason over all interrelated variables.

3.4 | Conclusion

Integrating AI in education marks a significant transition towards more interactive and personalised learning experiences. This comprehensive review illustrates AI's extensive impact on various educational aspects, from enhancing student learning and teaching strategies to streamlining assessment and administrative processes. The section on LLMs in education showcases explicitly their ability to generate relevant and pedagogically sound content, enriching the educational experience. Finally, we highlight the value of intelligent agents within the education field.

4

Methodology

In this chapter, we detail the steps taken to design, implement, and test the POC used in the conducted experiments. We begin by presenting the developed POC, a virtual educational environment referred to as Class Simulation. This environment comprises three core components: the Orchestrator, Student Agent, and Analyser. Each component is described in detail, including its role within the overall system. We then outline the process of selecting an appropriate LLM, collecting education-related data from experienced teachers describing classroom scenarios, and fine-tuning the selected model using this data. Finally, we evaluate the system's realism, pedagogical soundness, and component integration through expert review, user feedback, and technical performance analysis.

4.1 | Class Simulation

This section details the design and development of the POC, referred to as Class Simulation. Class Simulation comprises of three components: the Orchestrator, the Student Agent, and the Analyser. The process in Class Simulation (as seen in Figure 4.1) is as follows: First, the Orchestrator generates a scenario related to disruptive behaviours in class which is displayed to the teacher for them to handle. Then, the teacher tries to solve the scenario by talking to the Student Agents. These Student Agents process the teacher's input, adjust their emotions based on the sentiment detected, and respond to the teacher. Finally, once the teacher finished handling the scenario, they receive feedback on their actions from the Analyser. The teachers are allowed to ask questions to the Analyser if any clarification is required through the Mentor. Within this section, we first explore the design of the system, looking at the architecture of the POC. We then detail the technical implementation, including how we implemented the aforementioned Class Simulation components, and any additional implementation aspects, such as the design of the 3D environment in Unity, error handling mechanisms, and other quality-of-life

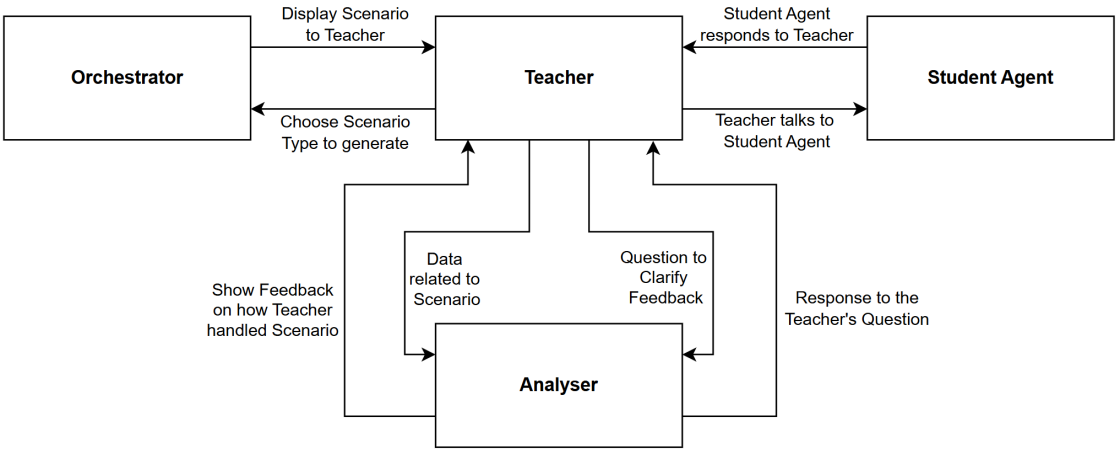


Figure 4.1: High Level view of the Class Simulation Architecture, including how the teacher interacts with each component. (Source: John Carmel Aquilina)

features integrated into the simulation. Finally, we detail our efforts in validating Class Simulation's functionality, along with the ethical concerns taken into consideration.

4.1.1 | System Design

This section details the design of Class Simulation. We go through each component's objective within the system and its' functionality pipeline.

4.1.1.1 | Orchestrator

The Orchestrator (as seen in Figure 4.2) is tasked with generating scenarios that include disruptive behaviour, which will then be handled by a teacher. The generated scenario includes: 1) the description of the scenario itself, and 2) the details of the student agents within the scenario, including their name, role in the scenario, introduction of the student in the scenario and their emotions.

The Orchestrator follows the below process:

1. First, the teacher selects a "general" scenario template which they wish to handle.
2. We then prompt a fine-tuned LLM to generate a scenario based on the chosen template. The model will respond with the description of the scenario, and a list of students who are present in the class.

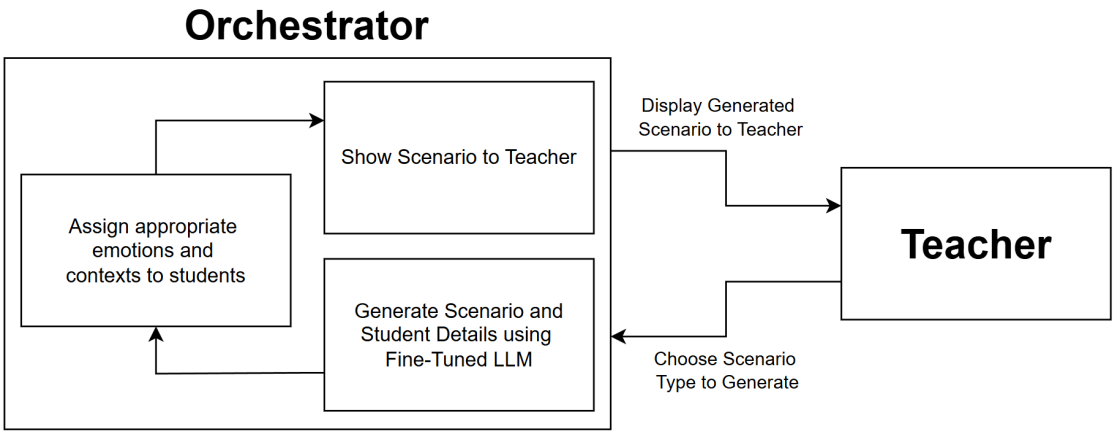


Figure 4.2: The design of the Orchestrator. Given a scenario type chosen by the teacher, the Orchestrator generates a scenario which is displayed to the teacher. (Source: John Carmel Aquilina)

3. The student list generated by the fine-tuned LLM is then assigned to the Student Agents. Each student contains various information, including their name, role, introduction to the scenario and initial emotions.

4. Finally, the generated scenario is displayed to the teacher.

4.1.1.2 | Student Agent

The Student Agent's (as seen in Figure 4.3) objective is to act as a student and react to the teachers' actions similar to how real life students would react. Several variables are essential within each Student Agent, including their name, role in the scenario, the introduction (what the student was doing before the teacher intervened), and the emotions of the student, which include: *upset*, *stressed*, *nervous*, *alert*, *excited*, *happy*, *sad*, *bored*, *relaxed* and *calm*.

The Student Agent follows the below process:

1. First, we extract the prevalent emotion and its intensity from the teachers' message to the student.

2. Then, we use the message and the detected emotion and its intensity to adjust the emotional values of the Student Agent.

3. Given the message, the new emotions, and who the teacher chose to talk to, the Student Agent chooses whether to listen to the teacher or respond to the teacher.

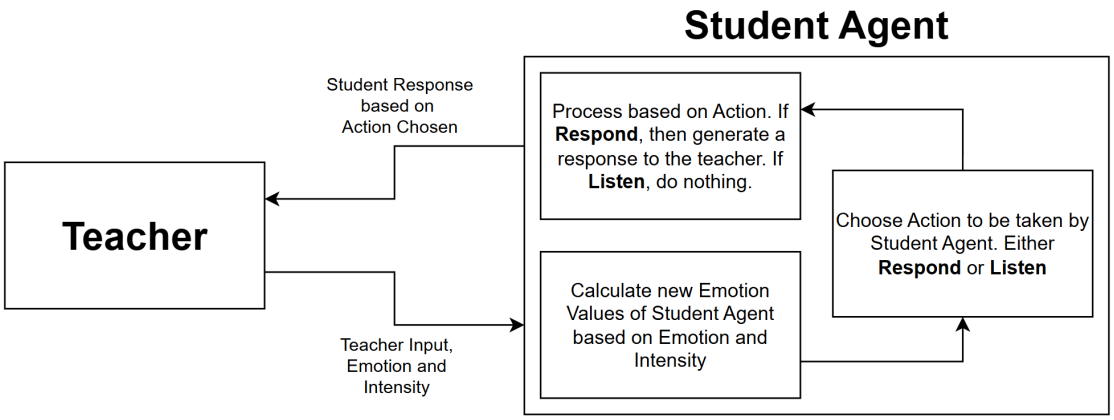


Figure 4.3: The design of the Student Agent. When the teacher talks to the Student Agent, its emotions are adjusted according to the emotion and intensity of the text, and generates a response based on the chosen action, which is then shown to the Teacher. (Source: John Carmel Aquilina)

4. If the teacher chose to speak to only one Student Agent, they will be forced to respond. This is to remove the possibility of the Student Agent not responding and misleading the teacher into thinking that the system is broken.

5. If the teacher chose to speak to none of the Student Agents, they will be forced to listen. This is to remove the chance of several Student Agents talking at the same time, making it difficult to understand what each of them said.

6. if the teacher chose to speak to more then one Student Agent, each Student Agent is prompted to choose whether to listen to the teacher or respond to the teacher.

7. If the Student Agent decides to listen, then the Student Agent does not generate a response.

8. If the Student Agent decides to respond, then the Student Agent generates a response directed to the teacher.

9. The result of the Student Agent's action is then shown to the teacher, with the teacher being able to take their next action based on the Student Agent's response.

4.1.1.3 | Analyser

The Analyser's (as seen in Figure 4.4) objective is to: 1) Analyse how the teacher handled the scenario and generate constructive feedback for the teacher, and 2) act as

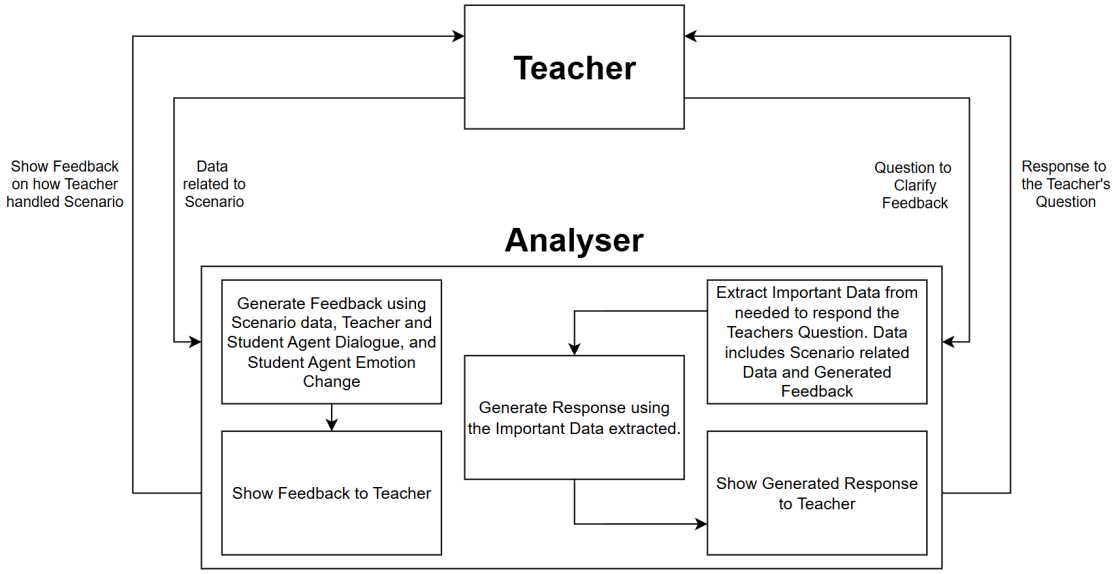


Figure 4.4: The design of the Analyser. Given the data related to the handled scenario (scenario description, teacher and Student Agent dialogue and emotion changes), the Analyser generates feedback to the teacher. The teacher can also ask for clarification on the feedback, in which given the question, the important data required to answer the question is extracted, and a response to the question is generated and shown to the teacher. (Source: John Carmel Aquilina)

a "chat assistant", answering any questions asked by the teacher related to the feedback provided and the scenario.

The Analyser follows the below process:

1. First, we feed the scenario description, the dialogue history between the teacher and Student Agents, and the actions taken by the Student Agents into a fine-tuned LLM. This LLM generates feedback which is then shown to the teacher.

2. Once the teacher reads the feedback, they can either: a) ask for clarification on the feedback, or b) return to the main menu.

3. If the teacher chooses to ask for clarification, then the teacher is allowed to ask the Mentor (a component within the Analyser) any questions related to the feedback.

4. Once the teacher asks a question, we prompt the same fine-tuned LLM containing the information related to the handled scenario and the generated feedback.

5. The response generated from the fine-tuned LLM is then shown to the teacher.

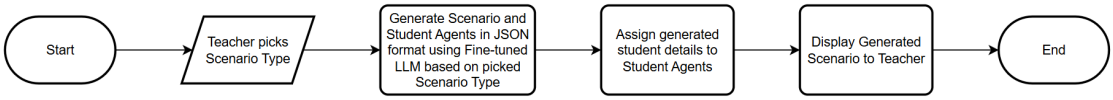


Figure 4.5: Orchestrator Flow Chart. (Source: John Carmel Aquilina)

6. The teacher can then either choose to continue asking question, or return to the main menu.

4.1.2 | Technical Implementation

This section outlines the technical development of Class Simulation. It covers the implementation of the core components described previously, and several supplementary features including the 3D environment created in Unity, error handling mechanisms, and other quality-of-life improvements. Additionally, we detail our efforts in validating the system's functionality, and address the ethical considerations inherent to this system.

4.1.2.1 | Orchestrator

To implement the Orchestrator, we create a process (as seen in Figure 4.5) which starts from the Teacher choosing a generic scenario type, and ends with the generated scenario being displayed to the teacher. The process is executed in the following order:

1. First, the teacher chooses a scenario type which they wish to handle. In our system, we provide teachers with three different choices: "*Student bullying another Student*", "*Two Students talking during a Lesson*", and "*Student damaging Property*".

2. The scenario type chosen by the teacher is used to prompt a fine-tuned GPT model. The model's settings are set as follows: Max Tokens is set to 2048, Top P is set to 1 and temperature is set to 1. We also define the response of the model, by setting it as a structured JavaScript Object Notation (JSON). The model is forced to include the following when generating a response: 1) Scenario Description which describes the scenario to be presented to the teacher, and 2) a Student Agent list which contains details on the students present in the classroom scenario. The list contains several details, including the name of the student, their role within the scenario, the introduction which describes the student's role in the scenario, and the initial emotions which assigns values from 0 to 1 to 10 emotions (upset, stressed, nervous, alert, excited, happy, sad, bored, relaxed and calm).

Figure 4.6: The student UI items which are updated to represent the student. Above the student are the Emotion Wheel and their name. When the orchestrator generates the Scenario, these are updated according to their assigned name and emotion values. (Source: John Carmel Aquilina)

3. The data generated in the Student Agent list is assigned to the Student Agents. This includes assigning their name, role, introduction and initial emotion values. These details will be used within the simulation both to display to the teacher which Student Agent they are talking to (as seen in Figure 4.6), and for the Student Agent to respond according to their assigned character.

4. Finally, we display the generated scenario in the 3D environment by displaying the scenario as an introductory text for the teacher. This generated scenario (as seen in Figure 4.7) is displayed to the teacher to provide them an introduction to the situation and the Student Agents involved.

To validate the implemented process is correct, we performed several tests:

- We tested the generation of the both the scenario and student details, confirming that the model produces a scenario description and student details that are consistent with each other. This was tested by prompting the GPT model and looking into the response it generates, and confirming that the details in the description match with the students details and assigned emotions. Relevant to this, we also tested that the

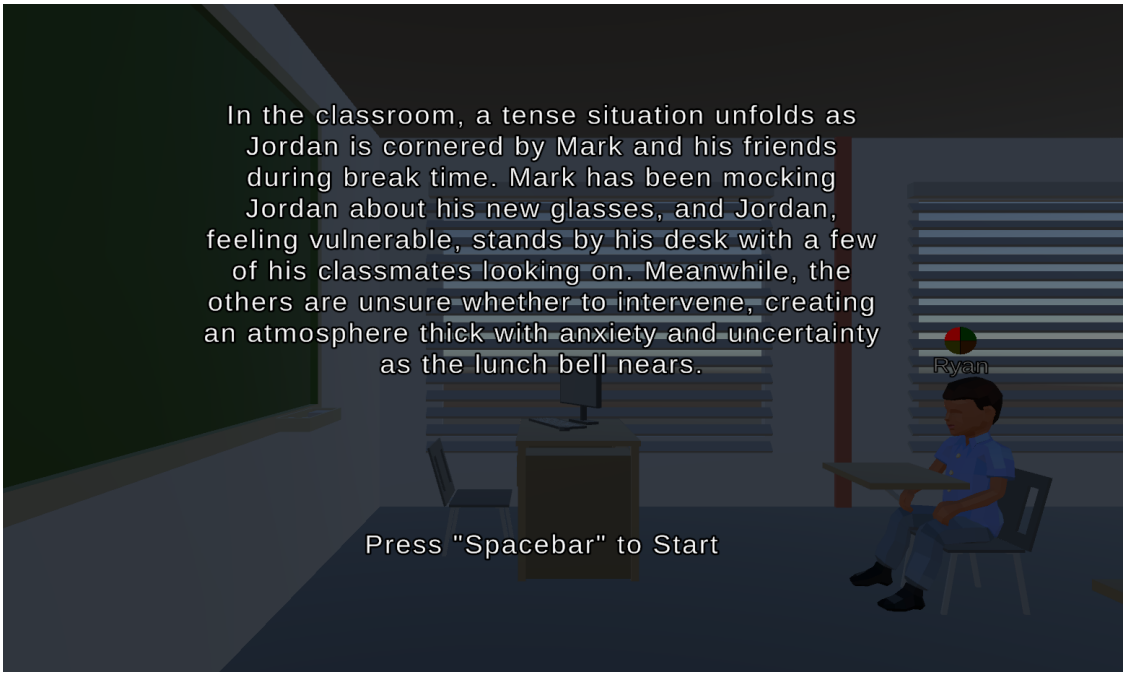


Figure 4.7: The Scenario generated using the Orchestrator which is displayed as text for the teacher. Once the teacher has understood the scenario, they can then press the "spacebar" to start interacting with the students. (Source: John Carmel Aquilina)

chosen scenario is represented in the generated scenario. This was also tested by prompting the GPT model and confirming the validity of the response.

- We tested the "assigning" of the details to the students and the scenario description within our system. This was tested by generating the scenario description and student details, and going through all the User Interface (UI) items and student agents and confirming that the scenario description and student details are assigned and displayed correctly.

- Finally, we tested the fine-tuning of the model and whether there was a difference between results generated from a normal model and a fine-tuned model. To test this, we prompted a normal GPT model and a fine-tuned GPT model to generate a scenario given the same scenario type. Given both responses and the fine-tuning data, we perform tests using BLEU, ROUGE, METEOR, and Cosine Similarity to measure the similarity of the generated text with the fine-tuning data. The results for these tests are further discussed in Chapter 5. Results & Evaluation.

4.1.2.2 | Student Agent

To implement the Student Agent, we define a process (as seen in Figure 4.8) which starts from the Teacher talking to the Student Agent, and ends with the Student Agent either responding to the teacher, or listening. This process is executed as follows:

1. First, the teacher talks to the student either by using their voice or by typing it. If voice is used, then we transcribe the audio using OpenAI's Whisper model.

2. Given the text, we analyse the emotion present within the text, along with its intensity, measured from 0 to 1. This is done by prompting a non-fine-tuned GPT model. The detectable emotions (taken from Hartmann, 2022's model) include *anger*, *disgust*, *fear*, *joy*, *sadness*, *surprise*, and *neutral*.

3. If the teacher is talking to a Student Agent, we adjust that Student Agent's emotions using the detected emotion and its intensity. We calculate the new value for the emotions using the following formula: $NewEmotionValue = OldEmotionValue + (DetectedIntensity \times AssignedMultiplier)$. The $OldEmotionValue$ is the Current Emotion value found in the Student Agent, $DetectedIntensity$ is the intensity of the emotion found in the previous step, and the $AssignedMultiplier$ is the multiplier assigned based on the detected emotion and which emotion within the Student Agent is being changed. A table of the $AssignedMultiplier$ values can be found in Table 4.1).

4. We then check whether the teacher is talking to multiple Student Agents. If they are only talking to one Student Agent, then we assign its action to **Respond**. If more then one Student Agent is selected, then we generate an action chosen using a non-fined-tuned GPT model. The possible actions include **Listen** or **Respond**. We provide the model with the Student Agent's details, their current emotions, and the teacher's input.

5. If the action chosen is **Listen**, then the Student Agent does nothing. If the action chosen is **Respond**, then we prompt a non-fine-tuned GPT model to generate a response to the teacher. The data used to generate the response includes the Student Agent's details (name, role, and intro), and the emotions of the activated quadrant along with how much they should affect the generated response (This can be found in Table 4.2).

		Emotion Detected						
		Anger	**Disgust**	**Fear**	**Joy**	**Sadness**	**Surprise**	**Neutral**
Emotion Effected	**Upset**	+0.5	+0.5		-0.5	+0.5		
	Stressed	+0.5	+0.5	+0.5	-0.5			
	Nervous	+0.5	+0.25	+0.5	-0.5		+0.25	
	Alert	+0.25		+0.25		-0.5	+0.5	
	Excited	-0.25	-0.5	-0.25	+0.5	-0.5	+0.5	
	Happy	-0.5	-0.5	-0.5	+0.5	-0.5		
	Sad				-0.5	+0.5		
	Bored		+0.25		-0.5	+0.5	-0.5	
	Relaxed	-0.5	-0.5	-0.5	+0.5	-0.25	-0.5	
	Calm	-0.5	-0.25	-0.5	+0.25	-0.25	-0.5	

Table 4.1: The $AssignedMultiplier$ based on the Emotion detected from the teacher's input, and the emotions they effect. (Source: John Carmel Aquilina)

		Unpleasant Activation			**Pleasant Activation**			**Unpleasant Deactivation**		**Pleasant Deactivation**	
		Upset	**Stressed**	**Nervous**	**Alert**	**Excited**	**Happy**	**Sad**	**Bored**	**Relaxed**	**Calm**
Response Effect	**Very High**	X									
	High		X			X					
	Moderate			X	X		X				
	Low							X		X	X
	Very Low								X		

Table 4.2: The Student Emotions used depending on the activated quadrant, along with how much they effect the Student Agent's generated response. (Source: John Carmel Aquilina)

6. Finally, the response generated by the Student Agent is converted into audio using Meta's TTS utilities. The audio is played to the teacher, along with a piece of text containing the generated response displayed on top of the Student Agent's head (as seen in Figure 4.9).

To validate that the Student Agent and its pipeline are correct, we performed several tests:

- We tested the transcription of the input and the detection of the emotion and intensity of the teacher input. We verified the transcription by speaking various statements and confirming that the transcription matches what we said. We then verified the detection of an emotion and its sentiment by inputting a variety of statements and noting which emotion was detected and how intense it is. We compared the results generated using our method to the results generated by (Hartmann, 2022)'s Emotion Classification model. This is further discussed in Chapter 5. Results & Evaluation.
- We tested the Emotion Adjustment by looking at the new emotion values after being affected by the emotion and intensity detected from the teacher's input and comparing them with our manual calculations of the new emotion values.

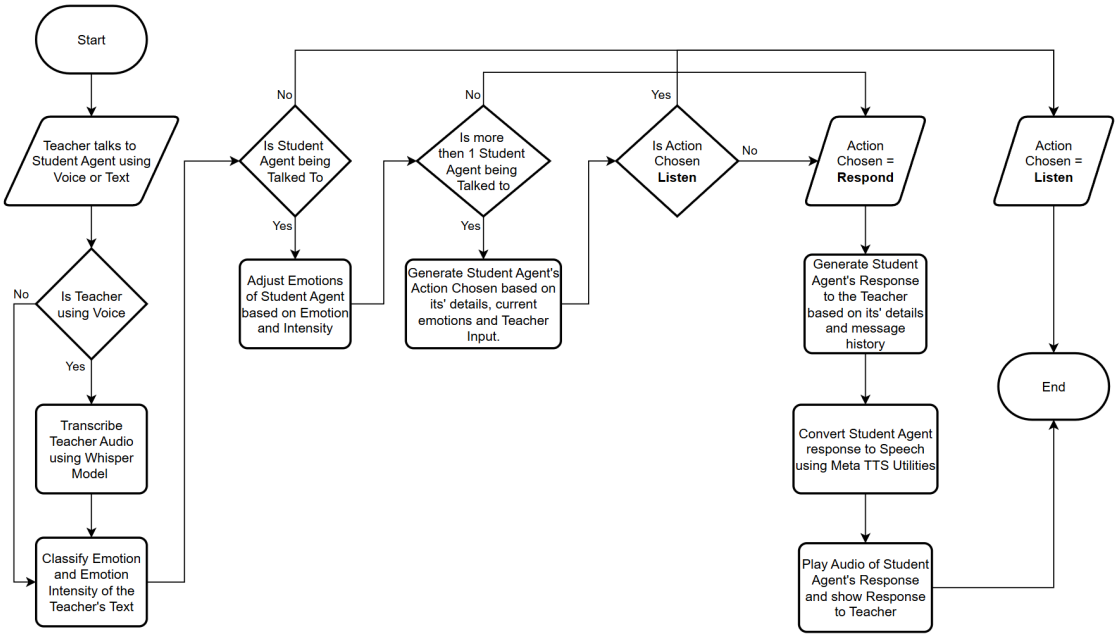


Figure 4.8: Student Agent Flow Chart. (Source: John Carmel Aquilina)

Figure 4.9: When the Student Agent generates a response, it is displayed as text above the Student Agent's head and played as audio using TTS. (Source: John Carmel Aquilina)

- We tested the Action Checks and GPT prompt to generate an action by activating each different check. First we selected a single student, and confirmed that the specific student always responds whilst the others don't respond. Then we selected two students, and confirmed that the two selected students prompt the GPT model to take an action and the remainder of the students don't respond. Finally, we didn't choose any of the students and confirmed that none of the students respond.
- Finally, we tested that the actions taken by the Student Agent are done correctly and displayed to the teacher. To test this, we first activated the "listen" action, in which the student agent does nothing. Then, we activate the "respond" action, which prompts a GPT model to generate a response for the Student Agent, and then shows it to the teacher through a text box on top of that specific Student Agents head and by playing the text as audio using TTS (The result can be seen in Figure 4.9).

4.1.2.3 | Analyser

For the Analyser, we define the following process (as seen in Figure 4.10) which is separated into two sections: 1) Gathering data from the handled scenario and using it to generate feedback for the teacher, and 2) generating a response to a teachers' question if they require clarification on the generated feedback. Both of these processes are executed as follows:

1. First, given the scenario, the message history between the teacher and Student Agents within the handled scenario, and the emotion changes of the Student Agents, we generate feedback using a fine-tuned LLM. This feedback is then shown to the teacher plain text (as seen in Figure 4.11).
2. If the teacher requires clarification on the generated feedback, they are allowed to ask questions to the Mentor (as seen in Figure 4.12).
3. To answer the question, we first extract the important data required to generate a response which answers the question. We do this by prompting a fine-tuned LLM, which then responds with two pieces of information: 1) whether any important information needed to answer the question was found, and 2) the important information required (if any information is needed) to answer the question.

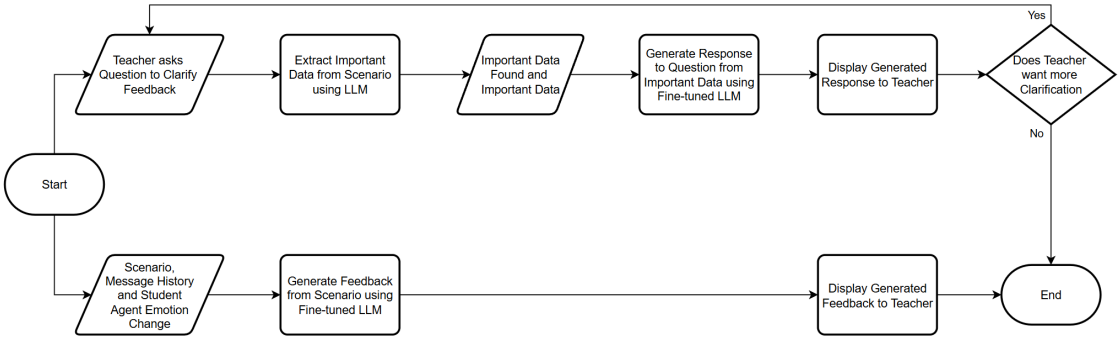


Figure 4.10: Analyser Flow Chart. (Source: John Carmel Aquilina)

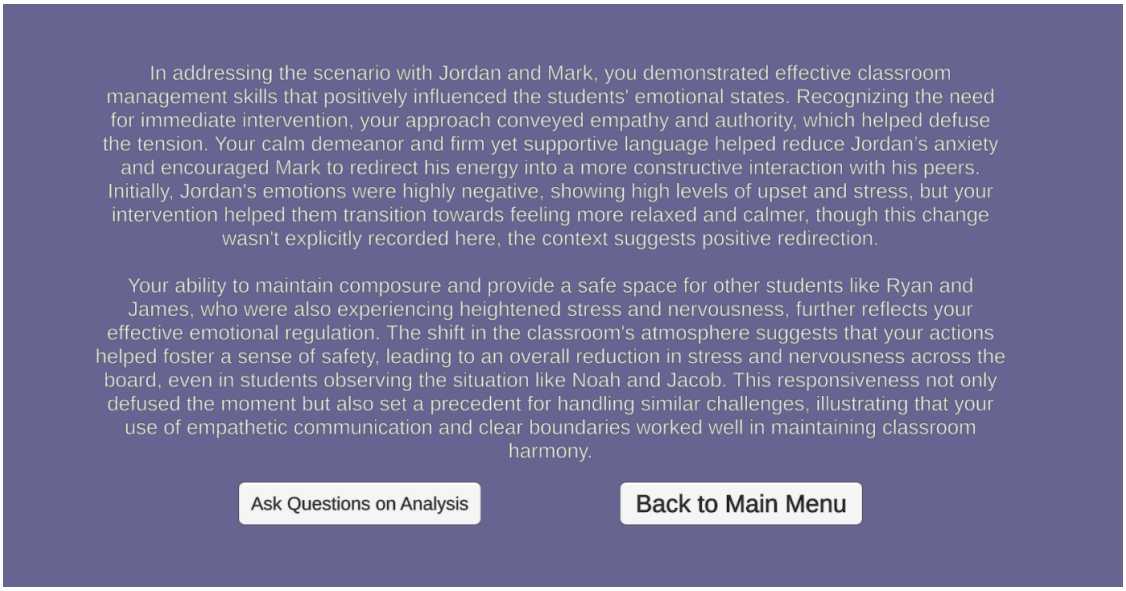


Figure 4.11: An example of how we show feedback on the teacher's handling of challenging behaviours within a scenario. (Source: John Carmel Aquilina)

4. We then prompt another fine-tuned LLM to generate a response to the teacher's question using the important information (if found in the previous step) to answer correctly.

5. Finally, we display the generated response to the teacher in text form. The teacher is then given an option to either continue or stop asking questions. If the teacher wishes to ask more questions, steps two to five are repeated.

To validate that the Analyser and its pipeline are correct, we performed several tests:

- We tested the data corroboration with the analysis generated by 1) reviewing the

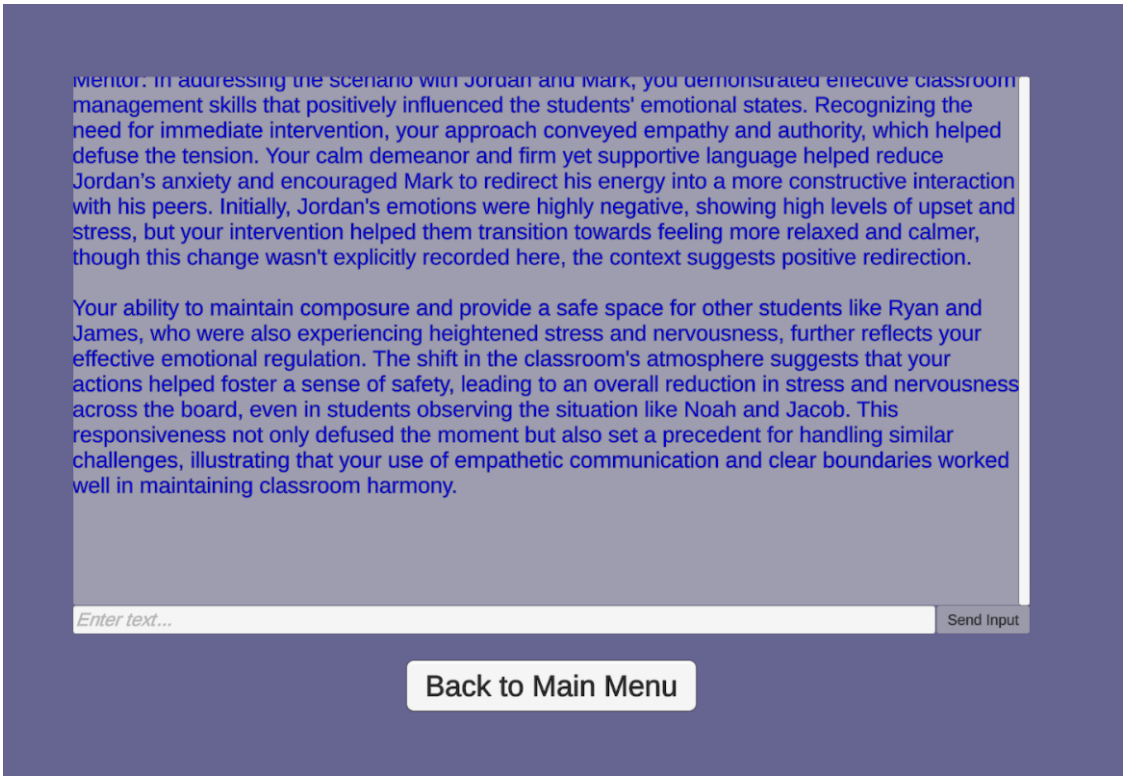


Figure 4.12: An example of how we allow the teacher to ask questions regarding the feedback provided to the mentor. (Source: John Carmel Aquilina)

generated feedback, and 2) comparing the analysis's feedback with the source data, thereby verifying the absence of hallucinations and accuracy of generated feedback.

- We tested the relevance of the important information extracted by 1) asking questions which require important information to generate a correct response, and 2) asking questions which are misleading, such as mixing up the student's name. Through these tests, we verify the relevance of the information extracted, both when the teacher asks a good question and when the teacher makes a mistake in the question.

- We tested the quality of the generated Mentor response by looking into 1) whether the response uses the important information extracted before, 2) whether the Mentor hallucinates information about the scenario, and 3) whether the Mentor corrects the teacher if they made a mistake in the question. By confirming the responses match the above checks, we ensure that the Mentor is generating correct responses.

- We tested that the generated response from the Mentor is displayed to the teacher

in the chat box. This was tested by asking a question to the Mentor, and confirming that 1) a response was generated and 2) the generated response is displayed for the teacher to read.

- Finally, we tested the fine-tuning of the model and whether there was a difference between results generated from a normal model and a fine-tuned model. To test this, we prompted a normal GPT model and a fine-tuned GPT model to generate an analysis. Given both responses and the fine-tuning data, we perform tests using BLEU, ROUGE, METEOR, and Cosine Similarity to measure the similarity of the generated text with the fine-tuning data. The results for these tests are further discussed in Chapter 5.

4.1.2.4 | Extra Implementations

In this section, we detail several implementations which greatly improved quality in our POC. These implementation range from the 3D Environment created in Unity to the handling of API calls used for prompt the GPT models.

Unity 3D Environment In Unity, we developed a 3D environment to allow teachers to interact with the Orchestrator, Student Agent and Analyser. In this section, I will be showing all of the menus built to facilitate easy communication between the teacher and components, as well as the 3D environment build to represent the classroom and the students.

Main Menu The main menu is the first menu teachers will see when they access Class Simulation. From this menu (seen in Figure 4.13), teachers can 1) select the scenario template they wish to handle in the class simulation (seen in Figure 4.14), 2) start the simulation after selecting the scenario, 3) go into the options menu to pick whether they prefer to use voice or text to talk to the students (as seen in Figure 4.15), and 4) quit the simulation. The main menu allows the teacher to communicate with the Orchestrator by picking a scenario type they wish to generate and starting the generation of the scenario by pressing the "Start Simulation" button. Once the scenario is successfully generated, the teacher will be automatically sent to the Virtual Classroom.

Virtual Classroom The virtual classroom is shown to the teacher once a scenario has been successfully generated. This environment allows the teacher to get acclimated

Figure 4.13: Menu items presented to teacher in the Main Menu (Source: John Carmel Aquilina)

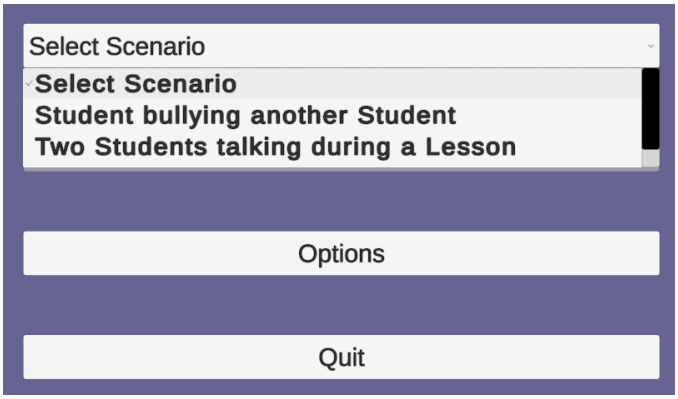


Figure 4.14: Dropdown with scenario types for the teacher to choose (Source: John Carmel Aquilina)

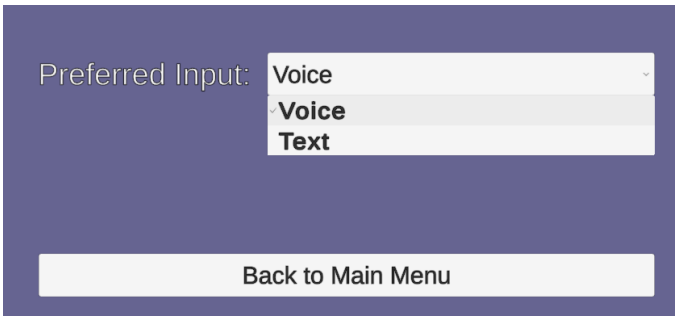


Figure 4.15: Options Menu with Preferred Input Dropdown (Source: John Carmel Aquilina)

Figure 4.16: Virtual Classroom which the teacher will use to interact with the Student Agents (Source: John Carmel Aquilina)

to the scenario generated from the Orchestrator, and it allows them to interact with the Student Agents. The first thing the teacher will see is the description of the generated scenario (as seen in Figure 4.7). Once the teacher has read the scenario description and decided to continue, they are then presented with the virtual classroom (as seen in Figure 4.16).

Within the virtual classroom, they are able to 1) select the students they wish to talk to by pointing the camera towards the student and pressing the "E" button (as seen in Figure 4.17), 2) talk to their students using voice or text (as seen in Figure 4.18), the input type is based on what they chose previously in the main menu, and 3) access the Pause Menu (as seen in Figure 4.19) to get help on the meaning of the Emotions Wheel (as seen in Figure 4.20) or end the class simulation and generate feedback on the teachers' actions. If the teacher decides to end the scenario, the data will be give to the analyser to generate feedback to the teacher, and the teacher will be sent to the Analyser Menu.

Feedback Menu The feedback menu is shown to the teacher when they decide to end the scenario. This menu allows the teacher to 1) look at the feedback provided by

Figure 4.17: Interact Student Example, when "E" is pressed, it will signify that the teacher is talking to that student (Source: John Carmel Aquilina)

Figure 4.18: Input methods which the teacher can use to interact with the Student Agents (Source: John Carmel Aquilina)

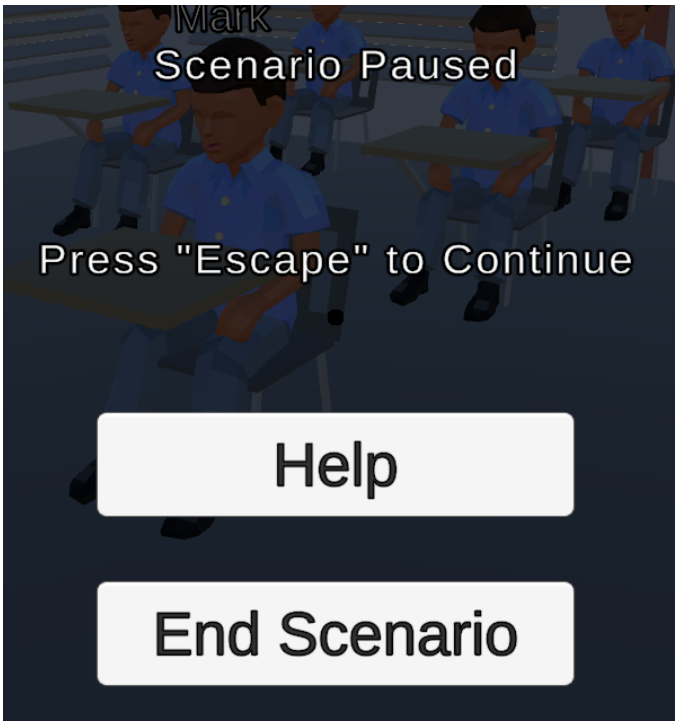


Figure 4.19: Pause Menu which the teacher can access by pressing the "Escape" Key (Source: John Carmel Aquilina)

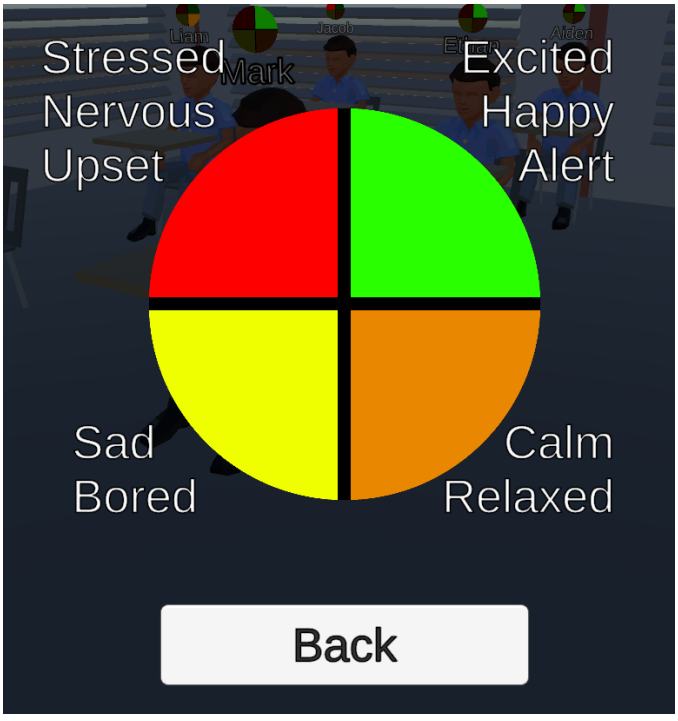


Figure 4.20: Help Menu which details which emotions correspond to each quadrant (Source: John Carmel Aquilina)

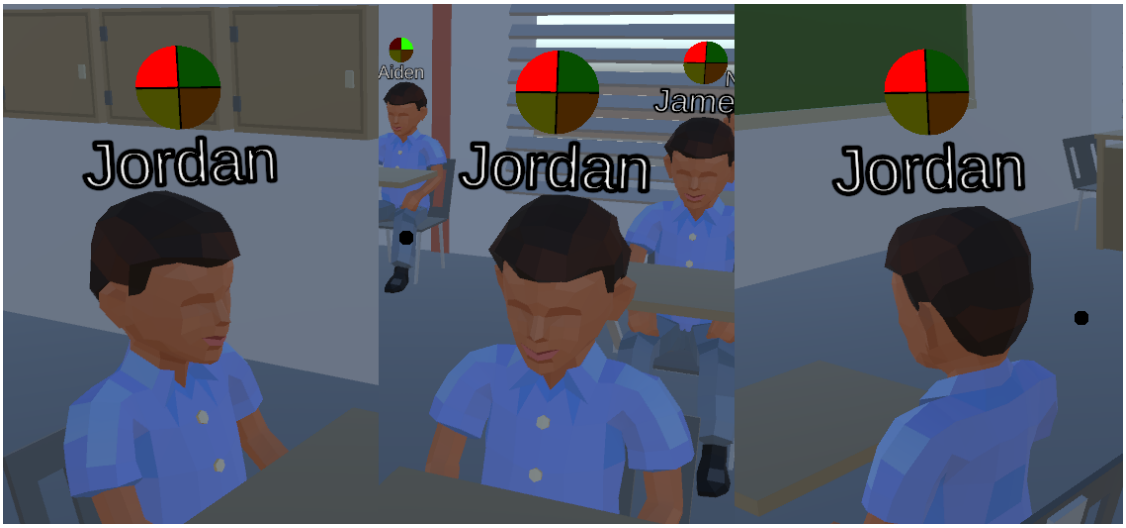


Figure 4.21: Using Floating Boxes, the Student Name and Emotion Wheel are always facing the Camera (Source: John Carmel Aquilina)

the analyser, and 2) ask questions for clarification in regards to the feedback if needed. The first thing the teacher will see is the feedback generated by the analyser (as seen in Figure 4.11). After reading the feedback, the teacher can then either ask for clarification on the feedback, or go back to the main menu. If the teacher decides to ask questions to the mentor, then they will be sent to the mentor menu (as seen in Figure 4.12), in which they can enter a question in the input field, and get a response from the mentor.

Interactor The interactor is a component built in Unity to allow the teacher to select which students they want to talk to. This component works by casting a ray from the player's perspective going forward and checking whether the ray is colliding with a GameObject containing the Interactor interface. If the GameObject being hit has the Interactor interface, then we outline the object and show a piece of text signifying that the current object is interact-able (as seen in Figure 4.17.

Floating Boxes The Floating Boxes is a component built in Unity which rotates UI objects to face the main camera whilst keeping it up-right. With this, regardless from which direction the teacher is looking at the student, the text will always be facing the camera (as seen in Figure 4.21).

OpenAI API calls Each section of the POC involved sending multiple API calls to OpenAI to interact with GPT models and transcribe audio recordings. Two distinct Unity

web requests were implemented to facilitate this: (1) transcription using the OpenAI Whisper-1 model, requiring binary audio data, and (2) prompting various GPT models (both fine-tuned and non-fine-tuned), necessitating specification of the model, prompt, maximum token count, and, where applicable, desired response format. The resulting OpenAI API responses were then processed to extract relevant information for the POC.

Audio Recording and Transcription To facilitate the interaction between the teacher and the Student Agent in our Class Simulation, we capture the teacher's audio and transcribe it through a two-step process: first, microphone input is recorded and saved as a Waveform Audio File (WAV) file upon button press, and second, the resulting recording is converted into binary data and sent to OpenAI's whisper-1 model via a Unity Web Request for transcription.

Text-to-Speech To ensure that a teacher sees the response generated by a Student Agent, we convert the text into speech, which is then played for the teacher. This was implemented using Meta's Voice Software Development Kit (SDK). Specifically, once the Student Agent generates a response, we process it through the TTS Speaker within the SDK, resulting in an audio output that is played for the teacher.

Error Handling To ensure experiment robustness, we incorporate error handling in Class Simulation. This allows the experiment to proceed uninterrupted when errors occur, specifically during: 1) interactions with the OpenAI API for GPT model prompting and audio transcription, and 2) calls to the Speak method during Student Agent response generation. Apart from handling the errors which could potentially occur whilst processing API calls, we also check for a valid internet connection before-hand to avoid any unnecessary issues related to not having a stable connection.

Status Information The status information text is used to signify what processing is currently occurring within the Class Simulation (as seen in Figure 4.22). More specifically, this is shown whilst the teacher's input is being transcribed and analysed, or when the student agents are processing the input and choosing an action. This was implemented with the purpose of easily noticing if Class Simulation freezes whilst prompting the GPT model.

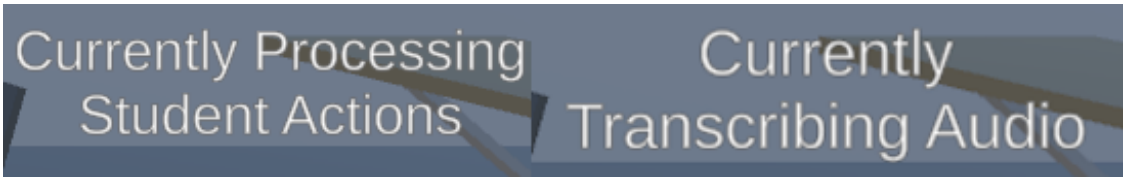


Figure 4.22: Status Information text used to show us whether the class simulation is currently processing a response. (Source: John Carmel Aquilina)

Saving Data Locally We locally save data generated from Class Simulation for subsequent analysis and validation. This includes: scenario descriptions and student data; messages sent within the simulation; emotion changes within the artificial agents; and messages sent between the mentor and teacher. This data, saved as plain text, will be used to analyse various metrics and validate the output of the models used within the simulation.

4.1.3 | Simulation Validation

The efforts which went into validating that the simulation works correctly can be split into two different sections:

- **Functionality**: We conducted unit testing to ensure that each component of the system, including the additional features we implemented, performed as expected. Additionally, we recruited independent testers (individuals not involved in the system's development) to pilot the simulation similarly to how the participants will be tasked to use it. This external testing helped in uncovering bugs and issues that might have been overlooked by us, thereby enhancing the overall robustness of the simulation.

- **Usability**: For usability validation, the independent testers who piloted the system were also asked to provide feedback on the simulation's UI, highlighting any aspects of the system that were unclear or confusing to use. Their insights enabled us to implement quality-of-life improvements that enhance the participant's experience and the system's usability. Additionally, this feedback informed us about the parts of the system which require further explanation or clarification during the experiment.

4.1.4 | Ethics Considerations

Ethical considerations are fundamental to the design and implementation of this system. This subsection outlines the key measures adopted to ensure user anonymity and secure the data collected during the experiments. Finally, we detail what should be done in the future to ethically implement Student Agents.

For user anonymity, we didn't have to change a lot to ensure that user data isn't tracked. This is mainly due to the fact that the participants utilised our devices to access the experiment, so there wasn't any personal data inputted by the participants to begin with. We also ensure that data is not associated with the participants by not asking for any information from the participants, and not matching any of the data with the participants. For participants who used voice feature to talk to the students, we also ensured that after the voice was transcribed, the file containing the voice data was not saved along with the anonymous data within the experiment, and that the file itself was deleted after the experiment finished.

Securing our data was not a huge concern due to the system being used locally, that being said we still take some precautions to be safe. Firstly, as mentioned above, the participants have to utilise our device to take part in the experiment, which removes the risk of participants personally accessing the data or data being unsecure on the participants personal device. Apart from the above, the researchers directly working on this dissertation are only allowed access to the data. Additionally, as mentioned before, we don't ask for any personal data from the users, which removes the risk of harm being done to the users if the data is somehow compromised. Finally, after the scope of this dissertation has been finished and the data analysed, all of the gathered data will be deleted correctly, ensuring that it is not recoverable.

Since this is a POC, additional measures need to be taken when implementing the Student Agents so that these adhere to ethical guidelines. An important measure which should be taken is to represent diversity in the classroom, with includes representing students of different genders, ethnicities, races, and backgrounds. This would help the teachers get acclimated to students and their varying cultures, informing them on how they should act depending on the background of the student. Correctly implementing this requires collaborating with teachers to analyse the different characteristics of students.

4.2 | LLM and Fine-Tuning

This section outlines the process of selecting a suitable LLM, gathering the classroom data from teachers, and fine-tuning the selected model with the purpose of using it in Class Simulation.

4.2.1 | LLM Selection

When deciding on which LLM to integrate into Class Simulation, several criteria were considered to ensure both technical feasibility and educational effectiveness. These criteria included 1) the performance of the model, 2) how compatible and easy it is to implement within the Unity game engine, 3) the level of support for fine-tuning with domain-specific data, and 4) the costs associated with fine-tuning and utilising the model.

1. **Model Performance**: The primary consideration was the baseline performance of the LLM. For this, we looked into the model's ability to 1) understand and generate responses which are coherent, 2) follow the rules set within the prompt, and 3) correctly act it's assigned role.

2. **Ease of Use in Unity**: Given that the POC was built using Unity, it was essential to select an LLM that could be integrated easily into the Unity game engine. This meant prioritizing LLMs which could be easily accessed through an API or a readily available Unity package.

3. **Fine-tuning Capability**: Since the simulation requires the components to generate responses that reflect realistic classroom scenarios and provide meaningful educational feedback, the ability to fine-tune the model on a specialised dataset is a major factor. Equally important was ensuring that the fine-tuning process has a tangible impact on the LLM, verifying that the model is adapted to the new data.

4. **Usage Costs**: Usage cost was another key factor in selected an appropriate LLM, particularly due to how frequent it is going to be prompted in the POC. Apart from looking into prompting costs, we also look into fine-tuning costs which typically includes a one-time training fee and higher usage rates for fine-tuned models.

Following the above criteria, we decided to integrate the GPT-4o mini into Class Simulation due to its balance between performance, cost, and ease of integration. Despite

	GPT-4o	GPT-4o (Fine-tuned)
Training Data	N/A	$3.00 / 1M tokens
Input	$0.15 / 1M input tokens	$0.30 / 1M tokens
Output	$0.075 / 1M output tokens	$0.15 / 1M tokens

Table 4.3: Comparison of usage and fine-tuning costs for the GPT-4o mini model. Costs are shown for standard prompting and for a fine-tuned version of the model, including training data, input, and output token rates.

being smaller compared to full-scale models, GPT-4o mini showed strong language understanding and generation capabilities. The model can be easily accessed within Unity by sending web requests to the OpenAI API using Unity's built-in `UnityWebRequest` methods. Fine-tuning any GPT models is easy through OpenAI's playground. The associated usage and fine-tuning costs are relatively low, making the model viable for sustained use in the POC — a breakdown of these costs can be found in Table 4.3.

4.2.2 | Classroom Data Collection from Educators

To collect data for fine-tuning, we developed an anonymous questionnaire with the purpose of gathering insight on teachers' personal teaching experiences. This questionnaire focused on teachers' observations of how specific behaviours and actions can influence students' emotions within a variety of classroom scenarios. The questionnaire consisted of several open-ended questions that asked teachers to describe positive and negative classroom scenarios, in particular those affecting student emotions. It also asked questions related to teacher actions and strategies in response to behavioural challenges, emotional regulation techniques, and the perceived impact of these behaviours on the mood of a classroom. Certain questions covered situations such as: waiting for a teacher to arrive, disruptions during lessons, and end-of-class transitions. Additionally, participants were asked to reflect on how their own emotional responses and professional development needs influence classroom management.

4.2.3 | Fine-Tuning the LLM

To fine-tune the LLMs used in the Orchestrator and Analyser components of the POC, we followed a structured three-step process:

1. We began by extracting the relevant data from the answers collected in the questionnaire outlined in Section 4.2.2. Specifically, responses to questions 4 and 7

```
{
"input":
    {
        "messages": [{"role": "user", "content": "Student bullying another
        Student"}],
        "tools": [],
        "parallel_tool_calls": true
    },
"preferred_output":
    [{"role": "assistant", "content": "A classroom is in session and one student is
    being bullied by another student. The bully is calling the other student names
    and shouting at him."}],
"non_preferred_output":
    [{"role": "assistant", "content": "A classroom is in session and one student is
    being..."},
    {"role": "assistant", "content": "A classroom is in session and one student is
    being bullied by another student..."},
    {"role": "assistant", "content": "This is about bullying, but lacks detail."}]
}
```

Figure 4.23: Example of a fine-tuning data-point used when fine-tuning the Orchestrator. (Source: John Carmel Aquilina)

were selected for fine-tuning the Orchestrator, while responses to questions 8, 9, 11, and 12 were used for the Analyser.

2. Using this data, we developed two separate fine-tuning datasets. For the Orchestrator, we defined 30 classroom scenarios, 10 examples for each of the three predefined situation templates (fine-tuning example found in Figure 4.23). For the Analyser, we created 10 feedback points, showing the ideal feedback to be given to a teacher based on their actions (fine-tuning example found in Figure 4.24). Additionally, for both fine-tuning datasets, we include examples for non-preferred responses to guide the model towards a more aligned behaviour.

3. Finally, the models were fine-tuned using Direct Preference Optimization (DPO), a method of fine-tuning offered by OpenAI that enables more efficient and effective alignment of LLMs with human preferences compared to traditional fine-tuning techniques. The hyperparamters defined when fine-tuning the models were 10 Epochs, Batch size = 1 and Learning Rate multiplier = 1. The hyperparameters were set to the default and not methodologically tested, with our focus being more on the integration of Class Simulation

With the size of the datasets being small as compared to typical datasets which include a minimum of 100 data-points, we recognise that this undermines the validity of the fine-tuning evaluation. Small datasets increase the risk of over-fitting, in which models memorise the training examples rather than learning generalisable patterns.

```
{
"input":
    {
    "messages":
        [{"role":"assistant","content":"Student: I don't want to work in a group. I hate working with them!"},
        {"role":"user","content":"Teacher: I understand that group work can be challenging. Can you tell me why you feel this way?"},
        {"role":"assistant","content":"Student: They never listen to my ideas."},
        {"role":"user","content":"Teacher: That sounds frustrating. What if we set some rules so everyone gets a chance to speak?"}],
    "tools":[],
    "parallel_tool_calls":true
    },
"preferred_output":
    [{"role":"assistant","content":"The teacher handled this situation well by acknowledging the student's frustration and prompting them to express their concerns. Instead of dismissing the student's feelings, they guided the conversation toward a constructive solution, demonstrating effective de-escalation and emotional regulation."}],
"non_preferred_output":
    [{"role":"assistant","content":"The teacher should have simply told the student that group work is mandatory and moved on. Addressing emotions in class wastes instructional time."}]
}
```

Figure 4.24: Example of a fine-tuning data-point used when fine-tuning the Analyser. (Source: John Carmel Aquilina)

4.3 | System Evaluation

To evaluate the developed POC, we conducted a series of tests. First, teachers participated in the Class Simulation experiment by using our system and completing the Usability Questionnaire, which incorporated questions inspired from Kriek and Stols (2010)'s study to measure the teacher's personal opinions, some questions from the Brooke (1996)'s SUS questionnaire, and a number of questions from Phan et al. (2016)'s GUESS questionnaire. Then, we designed a Validation questionnaire, where educational experts evaluated which generated responses from the fine-tuned models were more realistic, and identified the emotions expressed in the Student Agent's replies. Finally, using the data from the validation questionnaire and the classroom experiments, we assess the impact of fine-tuning using BLEU, ROUGE, METEOR, and Cosine Similarity, and we test whether the Student Agent emotions when generating responses matches the emotions assigned in the Validation questionnaire and the emotions detected using Hartmann (2022)'s emotion classifier.

4.3.1 | Usability Questionnaire

To gain insights into participants' perceptions of the system and its components, we developed a Usability Questionnaire. The questionnaire is divided into four sections, each designed with a specific focus.

1. **Technology Acceptance**: This section includes questions formatted after Kriek and Stols (2010)'s guide to examine the influence of beliefs of teachers on interactive simulations in their classrooms. This is split into three sub-sections: (1) Perceived Usefulness, (2) Perceived Compatibility and (3) General Technology Proficiency. Each question uses a 7-point Likert scale.

2. **Class Simulation Usability**: This section adapted several questions from Brooke (1996)'s SUS questionnaire, a well-established tool for evaluating system usability. We re-worded six questions from the SUS questionnaire to focus on assessing the usability of Class Simulation, each using a 5-point Likert scale.

3. **Class Simulation Engagement**: This section also adapts several questions from Phan et al. (2016)'s GUESS, a questionnaire which measures the players' satisfaction and user experience in video games. We adapt this to try and measure how engaging Class Simulation is, with each question using a 5-point Likert scale.

4. **Class Simulation General Feedback**: This section provides teachers with an opportunity to report any issues they encountered and to suggest improvements for Class Simulation.

4.3.2 | Validation Questionnaire

To ensure the content generated by Class Simulation is both realistic and feasible, we designed a Validation Questionnaire. We divide this questionnaire into three sections, with each section focusing on validating a specific component of the POC.

1. **Orchestrator Validation**: In this section, participants review pairs of generated scenarios and select the scenario they believe is more likely to occur in an actual classroom setting. For each pair, one of the scenarios was generated by the Orchestrator, and the other by a non-fine-tuned GPT-4o mini.

2. **Student Agent Emotion Validation**: Participants evaluate a series of responses generated by the Student Agent, assigning intensity scores (from 1 to 100) to a set of emotions, including **Sadness**, **Surprise**, **Joy**, **Disgust**, **Anger**, **Fear**, and **Neutral**. If an emotion is not evident in the response, they are instructed to leave the corresponding value at zero.

3. **Analyser Validation**: Following a similar format to the Orchestrator Validation, participants compare pairs of feedback messages given to teachers and indicate which feedback they consider more helpful and realistic. For each pair, one piece of feedback is generated by the Analyser and the other is generated by the non-fine-tuned GPT-4o mini.

4.3.3 | Fine-tune Similarity Tests

To test whether the fine-tuned models generated responses more similar to the fine-tuning data compared with responses from the non-fine-tuned models, we utilised four metrics: **BLEU** (Papineni et al. (2002)), **ROUGE** (Lin (2004)), **METEOR** (Banerjee and Lavie (2005)), and **Cosine Similarity**. These metrics were taken from Lakatos et al. (2025)'s paper in which they compared the performance of FT and Retrieval-Agumented Generation (RAG) models when implementing domain adaptation for developing Generative Large Language Model (G-LLM)-based knowledge systems.

4.3.4 | Student Agent Emotion Classification

To test whether the Student Agent's responses generated match their emotions, we compare the emotions assigned to the Student Agent before generating the response, the emotions assigned by the educational experts in the Validation questionnaire, and the emotions assigned by Hartmann (2022)'s `emotion-english-distilroberta-base` model. This model predicts Ekman's 6 basic emotions, plus a neutral class, of a piece of text. These six emotions include **anger**, **disgust**, **fear**, **joy**, **neutral**, **sadness**, and **surprise**.

4.3.5 | Questionnaire Ethics

When designing the questionnaires used for both experiments, it was important to ensure that they were implemented ethically. We took several measures to ensure that the

questionnaires were ethical:

- **Informed Consent Forms**: For the Usability Experiment, we gave each participant an informed consent form before the experiment, detailing the process of the experiment, and any necessary details. They were also asked for consent within the Usability questionnaire itself. For the Validation Experiment, we gave participants an introductory paragraph detailing what we are asking from them. Within the Validation questionnaire, they were asked for their consent, and the sections of the questionnaire were also explained.
- **Anonymisation**: For the questionnaires in both experiments, we made sure that the participant was kept anonymous by allowing anyone with the link to the Google Form to access the questionnaires without the need to login to their Google account. To take it a step further, educators who participated in the Usability experiment answered the questionnaire on our device, therefore ensuring that their data is not stored.
- **Data Retention**: We ensured the participants that data will not be retained once the scope of the research paper has been reached. Once the data has been analysed and discussed in the paper, it will be removed safely and without any risk of unauthorised personnel recovering and using it.
- **Extra Information**: Within the forms provided during both experiments, the participants were informed that their participation is voluntary and that they may opt out without any consequences. This is to ensure that participants do not feel pressured into participating or finishing the experiment. During the Usability experiment, participants were also given time to read the Informed Consent form provided, ensuring that they understood exactly what they are going to be participating in and that they are comfortable with participating.

4.3.6 | Participant Selection

When selecting participants for the experiments, we selectively sampled the educators to fit the needs of evaluating our POC.

- For the Usability experiment, we aimed to gather educators with various roles, such as Learning Support Educator (LSE)s, teachers, tutors without any teacher

training, and teachers pursuing a teaching related course. This would help us determine which roles within the educational field would benefit from our system. As a sample size, we aimed to gather ten participants.

- For the Validation experiment, a group of experts in pedagogy, in particular those who have direct experience with challenging classroom situations and behaviours were recruited. This was done to determine whether the fine-tuned models generate scenarios which are more realistic, and provide feedback which is more helpful. As a sample size, we aimed to gather five participants.

To recruit these educators, we contacted them through email. Participants were kept anonymous and their participation was voluntary. They were also allowed to opt-out during the experiment if they wished. The participants were not offered any compensation for participating.

4.4 | Conclusion

In summary, this chapter outlined the process of developing the POC, from selecting and fine-tuning the LLM to designing and validating Class Simulation. Through the integration of the Orchestrator, Student Agent, and Analyser components, we develop a system designed to help teachers improve their classroom management and handling skills in real-life scenarios. The evaluation process, involving expert reviews, user feedback, and technical assessments, provide valuable insights into the system's strengths and areas for improvement. The findings from this chapter establish a strong foundation for the subsequent analysis and discussion of the simulation's educational potential.

5

Results & Evaluation

In this section, we review the data generated from our experiments and how we evaluated it. We split this process into three sections. The Evaluation discusses how we evaluated Class Simulation, and the results of the evaluation. In the Results we discuss the results gathered from the experiments, going into detail on what the results mean. Finally, in the Discussion of Findings, we relate the results back to the research questions defined in Chapter 1.

5.1 | Evaluation

In this section, we detail the tests performed on the data gathered from the Usability and Validation experiments. We also look into the results of the evaluation and what it tells us about the implemented systems. Finally, we look back at the defined research questions, and analyse whether our implementation answers all of the specified questions.

5.1.1 | Evaluation Methods

There are several methods which we use to evaluate our POC. This section goes through the methods utilised to evaluate the feasibility of the simulations' components, the opinions of the participants taking part in experiments, and the usability of Class Simulation.

Class Simulation was evaluated in two separate steps. The first step is the Expert evaluation, in which we asked educational experts to fill in our validation form. This step was inspired from Shi et al. (2025)'s evaluation method of EducationQ, in which they ask expert reviewers to analyse teaching cases and validate the evaluator-agent-based qualitative methodology. Within our study, we ask the educational experts to validate the components as follows:

- **Orchestrator**: We provide the experts with two scenarios generated based on a single "template" scenario (for example, "*Student bullying another Student*"). One

of the scenarios is generated by our Orchestrator component which utilises a GPT-4o mini model fine-tuned on data gathered from teachers, and the other scenario is generated by a non-fine-tuned GPT-4o mini. For each pair of scenarios, the experts are asked to choose the scenario most likely to occur in a real-life classroom. The experts are also given a choice to pick "Both", signifying that both scenarios are as likely to occur, or "Neither", signifying that neither scenario is likely to occur.

- **Student Agent**: We provide the experts with several responses generated by the Student Agents. These experts are then asked to assign scores from zero to one hundred to the generated text based on how prevalent they feel the emotions are. The emotions used were taken from Hartmann (2022)'s `emotion-english-distilroberta-base` model, which classifies the emotions of a piece of text. These emotions include **anger**, **disgust**, **fear**, **joy**, **sadness**, **surprise**, and **neutral**.
- **Analyser**: Similarly to the Orchestrator validation, we provide the experts with two pieces of feedback generated based on what the teacher said to the Student Agents. One of the pieces of feedback is generated by our Analyser component which utilises a GPT-4o mini model fine-tuned on data gathered from teachers, and the other piece of feedback is generated by a non-fine-tuned GPT-4o mini. For each pair of feedback, the experts are asked to choose which generated response is the most helpful. They are also given a choice to pick "Both" if both pieces of feedback were equally helpful, or 'Neither", if neither of the feedback were helpful.

The second step is the Metric evaluation, in which we evaluate the components using a variety of metrics and models, and then compare the result of the Expert evaluation with the results gained from the Metric evaluation. Two sets of metrics were calculated for this step. To compare the results generated by the Orchestrator and Analyser with the non-fine-tuned GPT-4o models, we use metrics taken from Lakatos et al. (2025)'s paper in which they compared the performance of FT and RAG models when implementing domain adaption for developing G-LLM-based knowledge systems:

- **BLEU** (Papineni et al. (2002)) which is typically used to measure machine translation performance by measuring the n-gram accuracy. BLEU counts how many n-grams of the generated text are found in the reference translation.
- **ROUGE** (Lin (2004)) is used to measure the performance of machine translation and text summarisation tasks. This metric measures recall, which means it counts

how many n-grams of the reference translation are found in the generated text. ROUGE is useful as it is better then BLEU at taking into account the meaning of the text.

- **METEOR** (Banerjee and Lavie (2005)) is used to measure the performance of machine translation, text summaries, and creative text formats. It measures the Recall, Precision, and word order compatibility.
- **Cosine Similarity** is used to measure the similarity of texts. For Cosine Similarity to be used, we need to convert the text into sentence or word vectors, then calculate the cosine similarity between the vectors.

To classify the emotions present in responses generated by the Student Agents, we used the `emotion-english-distilroberta-base` (Hartmann (2022)) model, which classifies emotions of a piece of text, and predicts Ekman's 6 basic emotions, plus a neutral class: **anger**, **disgust**, **fear**, **joy**, **sadness**, **surprise**, and **neutral**. This model is a fine-tuned checkpoint of the `DistilRoBERTa-base` model, with various datasets (Demszky et al. (2020); Mohammad and Bravo-Marquez (2017); Mohammad et al. (2018); Poria et al. (2019); Saravia et al. (2018); Scherer and Wallbott (1994)) being used to train the model. These datasets contain emotion labels for diverse text types coming from Twitter, Reddit, student self-reports, and utterances from TV dialogues. The results of the metrics calculated are then compared to the Expert evaluation to confirm whether (1) the fine-tuned models used in the Orchestrator and Analyser generate better results then the non-fine-tuned version of the model, and (2) the emotions classified by the model matches the emotions assigned by the educational experts, and the emotions used within the Student Agent to generate the response.

To evaluate the personal opinions of teachers on the use of interactive simulations within the Usability questionnaire, we look at Kriek and Stols (2010)'s study. In their study, they create the Combined Model (as seen in Figure 5.1) which combines: (1) Theory of Planned Behaviour which explains human action and suggests that it is guided by behavioural, normative, and control beliefs, (2) Technology Acceptance Model, which attempts to explain factors that influence users' acceptance of information technology systems, and (3) Innovation Diffusion Theory, which is used to study a variety of innovations. We select three variables from the Combined Model, and create questions based on the variable: (1) **Perceived Usefulness** which measures how useful the participants

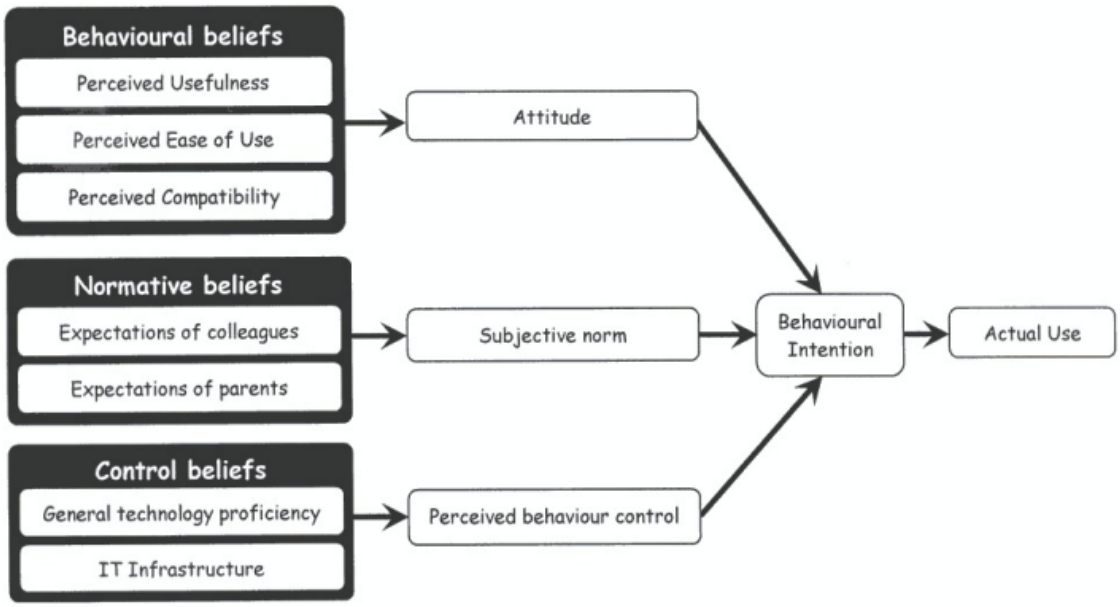


Figure 5.1: The Combined Model (Source: Kriek and Stols (2010))

perceive our Class Simulation, (2) **Perceived Compatibility** which measures how comparable our Class Simulations are to real-life classrooms, and (3) **General Technology Proficiency** which measures the teachers capabilities in using AI systems in educational scenarios.

To evaluate the Class Simulation system, we looked at two questionnaires to measure (1) the usability of the system, and (2) the engagement of the simulations. To measure the usability of the system, we adapted questions from the SUS (Brooke (1996)) questionnaire. The SUS questionnaire is a ten-item attitude Likert scale which gives a global view of subjective assessments of usability. To measure the engagement of the Class Simulation, we adapted questions from the GUESS (Phan et al. (2016)) questionnaire. The GUESS questionnaire is a psychometrically validated gaming scale with nine subscales. While the questionnaire was originally designed for play-testing and game evaluation, in this study we selected and adapted its questions to assess how engaging participants found Class Simulation's generated scenarios and Student Agents.

5.1.2 | Evaluation Results

The results from the methods described above will be separated into three sections. First, we will go through the results gathered from the validation of Class Simulation,

including going through the responses of the Validation form, and the results of the calculated metrics. Then, we will look into the result of the participant's personal opinions regarding AI and Class Simulation. Finally, we will look into the results of the usability and engagement of Class Simulation, and any feedback given by participants on what could be improved in the system.

5.1.2.1 | Expert and Metric Validation Results

This section presents the results obtained from the Expert and Metric evaluation for each component. For the Expert evaluation, we managed to find six experts willing to answer our questionnaire. When calculating the BLEU score for the Metric evaluation, we used smoothing method one which adds epsilon counts to precision with zero counts.

We begin with the Orchestrator results. For each pair of responses, Orchestrator is the response generated from the Orchestrator and GPT-4o mini is the response generated from the non-fine-tuned model. The results of the Expert evaluation (as seen in Figure 5.2) varies across all questions. For question 1, both responses were preferred equally, with the most popular option being "Both", and the Orchestrator and the non-fine-tuned GPT-4o mini model being picked equally. For question 2, the experts indicated that both generated scenarios were equally likely to occur in a real-life classroom, with a slight preference towards the Orchestrator's generated scenarios. For question 3, the experts preferred the non-fine-tuned GPT-4o mini model, with the second most popular option being that both of them generated realistic scenarios. The only common response between all three questions, is that none of the experts thought that neither of the scenarios generated were similar to real-life classrooms. From the Metric evaluation (as seen in Table 5.1), we get varying results when comparing the responses generated by the Orchestrator and GPT-4o mini with the fine-tuning data used in the Orchestrator. Looking at the BLEU score, the Orchestrator scored better on the prompts in questions 1 and 3, but performed slightly worse in question 2. For the ROUGE scores, the Orchestrator consistently achieved a higher Recall value then the GPT-4o mini model. The Precision and F-Measure scores were also consistently higher for the Orchestrator, with a few exceptions. These exceptions include the Precision of the GPT-4o mini model for question 3 in the ROUGE-1 and ROUGE-L metrics, as well as the Recall of the GPT-4o mini model for question 3 in the ROUGE-1 metric. For the METEOR score, the Orchestrator scored slightly higher across all of the prompts. For Cosine Similarity,

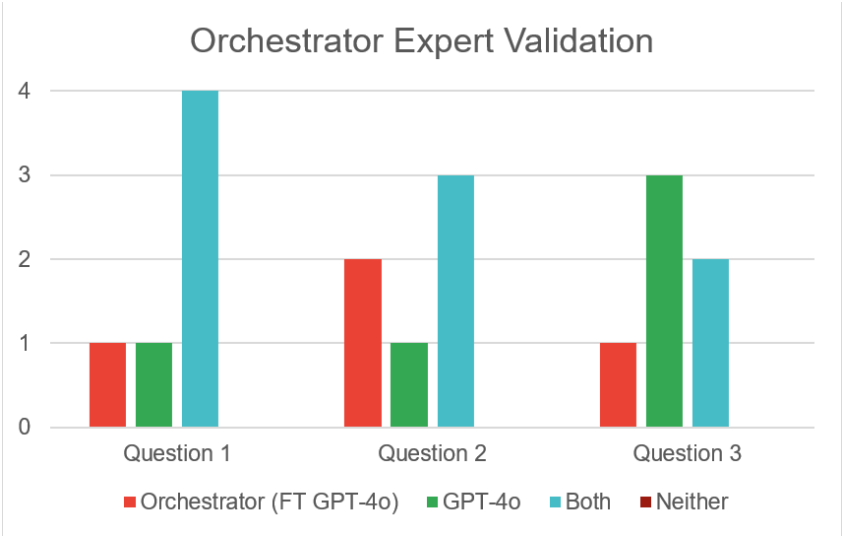


Figure 5.2: Bar Chart of the result of the Expert validation on the Orchestrator. (Source: John Carmel Aquilina)

		Question 1		Question 2		Question 3	
		Orchestrator	**GPT-4o mini**	**Orchestrator**	**GPT-4o mini**	**Orchestrator**	**GPT-4o mini**
BLEU	**Score**	**0.1675**	0.0331	0.2306	**0.2496**	**0.1524**	0.1109
ROUGE-1	**Precision**	**0.2674**	0.2262	**0.2849**	0.2630	0.2220	**0.2562**
	Recall	**0.1719**	0.1059	**0.1589**	0.1092	**0.1240**	0.1233
	F-Measure	**0.2059**	0.1420	**0.2010**	0.1521	0.1568	**0.1642**
ROUGE-2	**Precision**	**0.0466**	0.0228	**0.0631**	0.0583	**0.0370**	0.0327
	Recall	**0.0270**	0.0085	**0.0301**	0.0227	**0.0199**	0.0135
	F-Measure	**0.0330**	0.0121	**0.0398**	0.0320	**0.0252**	0.0186
ROUGE-L	**Precision**	**0.2525**	0.2012	**0.2528**	0.2277	0.2037	**0.2133**
	Recall	**0.1614**	0.0948	**0.1411**	0.0954	**0.1132**	0.1033
	F-Measure	**0.1937**	0.1267	**0.1783**	0.1325	**0.1433**	0.1373
METEOR	**Score**	**0.3838**	0.3341	**0.4270**	0.4010	**0.3171**	0.2956
Cosine Similarity	**Score**	0.1285	**0.1418**	**0.1665**	0.1479	0.0238	**0.1146**

Table 5.1: Metric evaluation results for the Orchestrator and GPT-4o mini across different prompts. These results include BLEU, ROUGE, METEOR and Cosine Similarity. (Source: John Carmel Aquilina)

the scenarios generated by GPT-4o mini in questions 1 and 3 performed better than the Orchestrator, and vice versa on question 2.

To analyse the results for the Student Agent's emotions, we compare the original emotions used by the Student Agent to generate the response being used in the question with both the emotions assigned in the Validation Evaluation and the emotion values classified using Hartmann (2022)'s `emotion-english-distilroberta-base` model. Due to an error encountered after the experiments, we removed any results with the **Surprise** and **Disgust** emotions from the emotion classification. For these results, we refer to the experts as E1 to E6. The original emotions used (as seen in Table 5.2) in the responses

	Stressed	Nervous	Upset	Sad	Bored	Calm	Relaxed	Excited	Happy	Alert
Response 1								X	X	X
Response 2	X	X	X							
Response 3	X	X	X							

Table 5.2: Emotions used by the Student Agent when generating a response to the teacher. (Source: John Carmel Aquilina)

provided to the experts are as follows: For response 1, the Student Agent was prompted to generate a response using the **excited**, **happy** and **alert** emotions, and for responses 2 and 3, the Student Agent was prompted to generate a response using the **stressed**, **nervous** and **upset** emotions. From the validation evaluation (as seen in Tables 5.3 and 5.4), we can see that most experts agreed which emotions were present in each response. For response 1, all of the experts agreed that **joy** was the most prevalent in the generated response. For response 2, five of the experts agreed that **sadness** was the most prevalent emotion in the generated response, whilst one expert disagreed with the majority, giving the highest score to **neutral**. Finally, for response 3, the experts collectively agreed that **anger** was the most prevalent emotion. Looking at the Metric evaluation which used the model from Hartmann (2022), the highest emotion scores assigned were as follows: Response 1 was predominantly associated with **joy**, receiving a score of 0.92; Response 2 was classified as **neutral** with a score of 0.67; and Response 3 was strongly linked to **anger**, with a score of 0.47.

Finally, we look into the results of the Analyser. For each pair of responses, Analyser refers to the response generated by the Analyser, and GPT-4o mini refers to the response generated by the non-fine-tuned model. From the results of the Expert evaluation (as seen in Table 5.3), both questions showed varying results. For question 1, experts found both pieces of advice to be equally realistic and consistent with guidance provided by real-life experts. For question 2, experts overwhelmingly found the advice given by the Analyser to be more realistic, with the number of experts choosing the normal GPT-4o mini model and both models being equal. What can be noted from these results is that, similarly to the Orchestrator's Validation evaluation, none of the experts thought that the responses were unrealistic. For the Metric evaluation (a seen in Table 5.6), we can see more varying results. Looking at the BLEU, the Analyser out-performed the GPT-4o mini in both questions. For the ROUGE scores, the Analyser almost always performed better than the GPT-4o mini in both questions, with a few exceptions. These exceptions include the F-Measure in ROUGE-2 and the Recall and F-Measure in ROUGE-L for

		Sadness	Anger	Fear	Joy	Neutral
Response 1	**E1**				**80**	
	E2				**100**	
	E3				**90**	
	E4				**95**	
	E5				**75**	
	E6				**80**	
Response 2	**E1**	**60**	20	10		
	E2	**70**				
	E3					**80**
	E4	**85**				
	E5	**80**				
	E6	**100**				
Response 3	**E1**	5	**80**			
	E2		**75**			
	E3		**90**			
	E4		**75**			
	E5		**100**			
	E6		**70**			

Table 5.3: Emotions classified by Experts (marked E1 - E6), the highest value given to an emotion is marked in bold. Each emotion can be scored from 0 to 100. (Source: John Carmel Aquilina)

	Sadness	Anger	Fear	Joy	Neutral
Response 1				**6**	
Response 2	**5**				**1**
Response 3		**6**			

Table 5.4: Counts of Experts identifying the most prevalent emotion for each response. (Source: John Carmel Aquilina)

	Sadness	Anger	Fear	Joy	Neutral
Response 1	0.00	0.01	0.00	**0.92**	0.02
Response 2	0.17	0.01	0.01	0.09	**0.67**
Response 3	0.02	**0.47**	0.01	0.01	0.08

Table 5.5: Emotions classified using Hartmann (2022)'s `emotion-english-distil roberta-base` model on the Student Agent responses. Each emotion is scored from 0 to 1. (Source: John Carmel Aquilina)

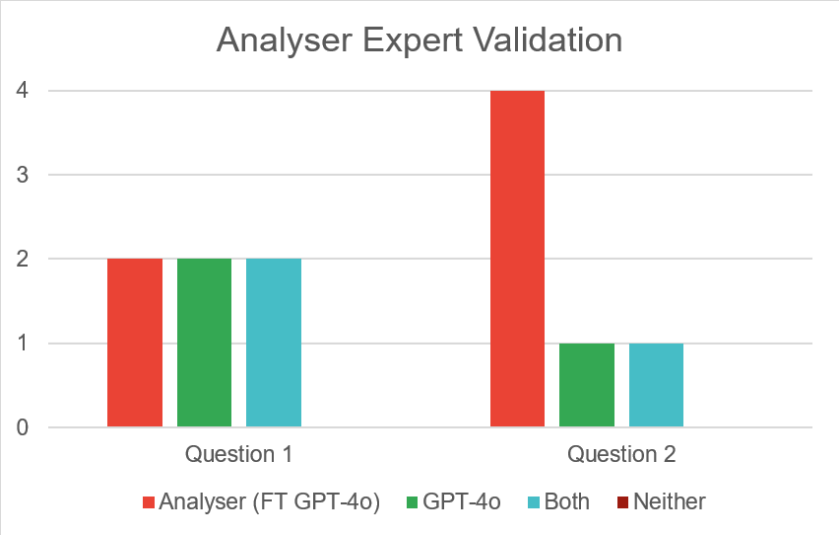


Figure 5.3: Bar Chart of the result of the Expert validation on the Analyser. (Source: John Carmel Aquilina)

		Question 1		Question 2	
		Analyser	**GPT-4o mini**	**Analyser**	**GPT-4o mini**
BLEU	**Score**	**0.0122**	0.0109	**0.0131**	0.0092
ROUGE-1	**Precision**	**0.3871**	0.3325	**0.3482**	0.3291
	Recall	**0.0711**	0.0705	**0.0669**	0.0616
	F-Measure	**0.1200**	0.1162	**0.1121**	0.1037
ROUGE-2	**Precision**	**0.0314**	0.0278	**0.0415**	0.0263
	Recall	0.0042	**0.0045**	**0.0061**	0.0036
	F-Measure	0.0074	**0.0077**	**0.0106**	0.0064
ROUGE-L	**Precision**	**0.3308**	0.3074	0.3123	**0.3153**
	Recall	0.0604	**0.0651**	**0.0599**	0.0589
	F-Measure	0.1020	**0.1073**	**0.1003**	0.0991
METEOR	**Score**	**0.2817**	0.2645	0.2426	**0.2826**
Cosine Similarity	**Score**	0.1770	**0.2185**	**0.1809**	0.1504

Table 5.6: Metric evaluation results for the Analyser and GPT-4o mini across different prompts. These results include BLEU, ROUGE, METEOR and Cosine Similarity. (Source: John Carmel Aquilina)

question 1, and the Precision for ROUGE-L in question 2. For the METEOR score, the Analyser got a higher score than the GPT-4o mini model in question 1, but a lower score in question 2. Finally, for Cosine Similarity, the Analyser performed worse than the GPT-4o mini model in question 1, but performed better in question 2.

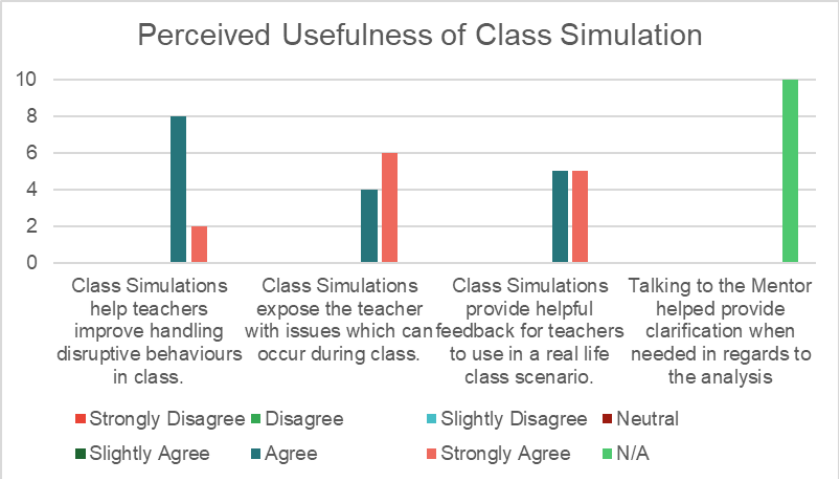


Figure 5.4: Bar Chart of the result of the Perceived Usefulness of Class Simulation. (Source: John Carmel Aquilina)

5.1.2.2 | Teacher Opinion Results

This section details the results on the Teacher's Opinions gathered from the Usability questionnaire. These results are separated into three sections: 1) Perceived Usefulness, 2) Perceived Compatibility and 3) General Technology Proficiency.

Perceived Usefulness Looking at the results (as seen in Figure 5.4), we notice two things. First, for questions one to three, the teachers agreed that each Class Simulation component can help in 1) improving teachers handling of disruptive behaviours in class, 2) expose teachers to issues which can occur during class, and 3) provide helpful feedback which can be used in real-life classroom scenarios. Second, we also note that all the feedback provided was clear enough that none of the participants needed to talk to the Mentor for clarification on the feedback.

Perceived Compatibility From the Perceived Compatibility section (as seen in Figure 5.5), participants mostly agreed that the Class Simulation generated: 1) Scenarios which are similar to real-life classroom scenarios, 2) Student Agent responses which are similar to how real-life students respond, and 3) feedback which is similar to feedback given by real life professionals. Specifically looking at the second point, one participants disagreed, and felt that the Student Agents did not respond similarly to real-life students.

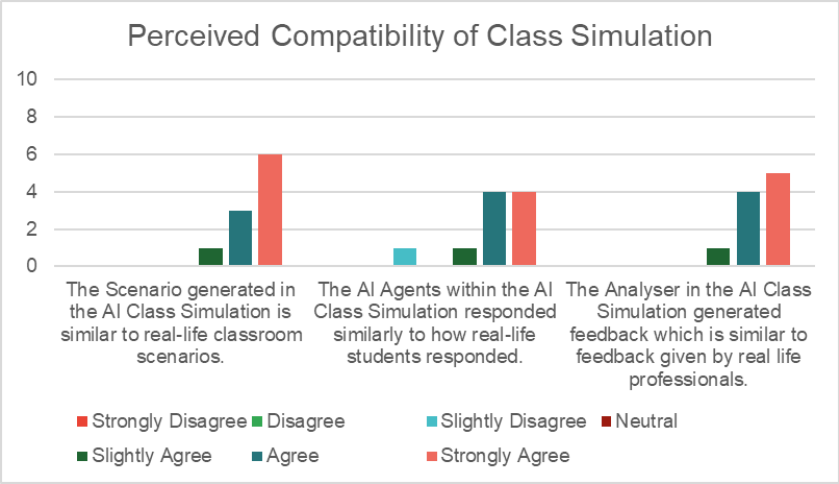


Figure 5.5: Bar Chart of the result of the Perceived Compatibility of Class Simulation. (Source: John Carmel Aquilina)

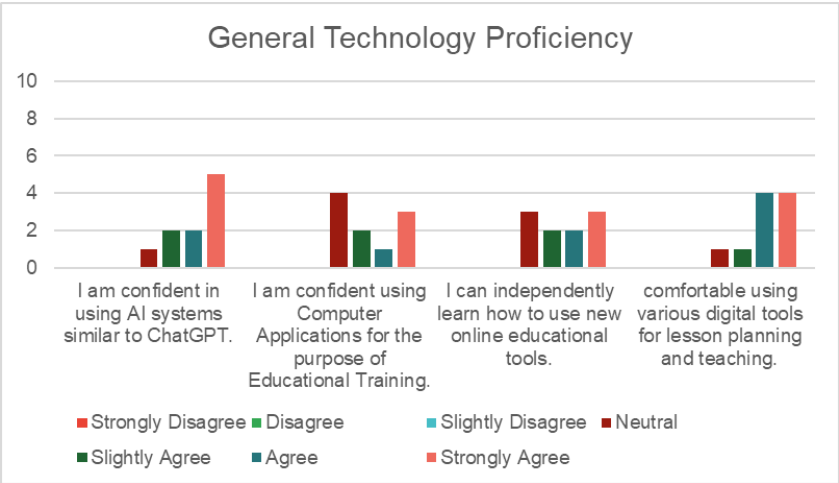


Figure 5.6: Bar Chart of the result of the General Technology Proficiency. (Source: John Carmel Aquilina)

General Technology Proficiency For General Technology Proficiency (as seen in Figure 5.6), we note that most participants agreed that they are confident in: 1) using AI systems similar to ChatGPT, 2) using computer applications for educational training, 3) independently learning how to use new online educational tools, and 3) using digital tools for lesson planning and teaching. Within these results, some participants answered with "Neutral", with questions two and three having the most participants who didn't agree.

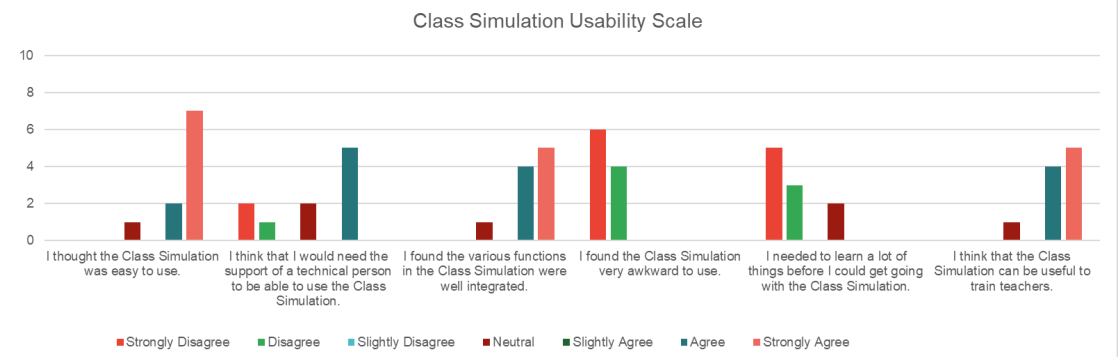


Figure 5.7: Bar Chart of the result of the Class Simulation Usability Scale. (Source: John Carmel Aquilina)

5.1.2.3 | Class Simulation Usability Results

The results presented in this section detail the participants opinion on the usability of Class Simulation. The results are separated into the Usability Scale and the Engagement Scale.

Usability Scale We detail several notes on the results gathered from the Usability Scale (as seen in Figure 5.7). First, participants mostly agreed that Class Simulation was easy to use, with one participants responding with neutral. For question two, responses varied with some participants disagreeing that they would need the support of a technical person to use Class Simulation, whilst others agreed that they would need it. Similarly to the first question, most participants agreed that the functions in Class Simulation were well integrated, except for one responding with neutral. Participants also overwhelmingly disagreed that Class Simulation was awkward to use, and also mostly disagreed that they needed to learn a lot before using Class Simulation (with two participants responding neutral). Finally, participants agreed that Class Simulation can be useful to train teachers, with one participant choosing neutral.

Engagement Scale The results gathered from the Engagement Scale (as seen in Figure 5.8) were mostly consistent throughout each question. For question one, most participants agreed that they were captivated by the simulation from the beginning, with two participants responding with neutral. All of the participants agreed that the characters in the simulation are well developed. For question three, most participants agreed that the characters in the simulation responded realistically, with one participant disagreeing. For questions four and five, most participants agreed that they 1) could clearly understand the

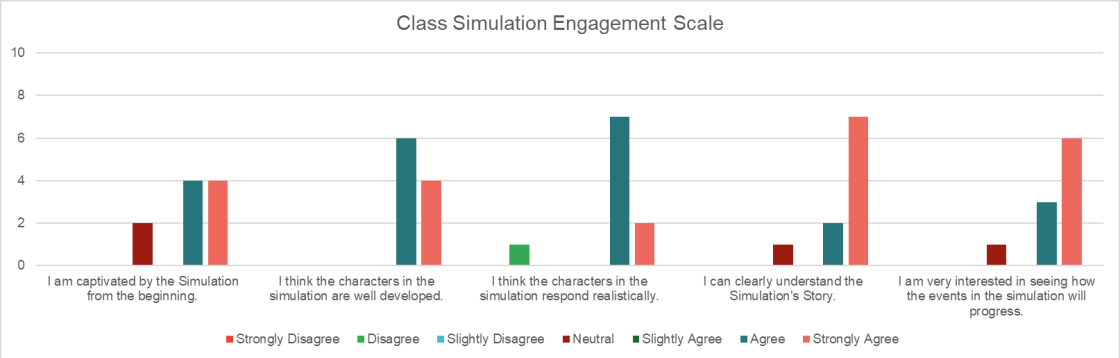


Figure 5.8: Bar Chart of the result of the Class Simulation Engagement Scale. (Source: John Carmel Aquilina)

simulation's story, and 2) were very interested in seeing how the events in the simulation will progress, with a participant in both questions responding with neutral.

5.1.2.4 | Participant Feedback

This section details the feedback related to Class Simulation written by the participants. This feedback will be split into three groups, 1) Student Agent responses, 2) system related issues, and 3) future improvements.

- **Student Agent Responses**: A main point of feedback regarding the Student Agent is that they were too agreeable and apologetic as compared to real life students. Although some students are receptive to feedback and advice, a lot of students take longer to listen, and aren't as apologetic as the Student Agents in the simulation. A participant also mentioned that the simulation students were introspective and understood their emotions, something which not all real life students have the capability of defining and regulating.

- **System Related Issues**: A reoccurring issue within Class Simulation is that when more then one student is being talked to and more then one student decides to respond, then the Student Agents will talk at the same time. Another issue which a participant commented on was related to the controls and how the E key, which was used to select the students the participant wanted to talk to, was difficult to use due to the "chat" functionality. With the chat being enabled by pressing the Enter key, sometimes the participants got confused as to whether they were typing to talk to the Student Agent, or selecting the Student Agent.

- **Future Improvements**: The participants proposed several improvements which could be implemented to Class Simulation. A re-occurring issue was the fact that teachers couldn't re-read the Student Agent's response due to it being removed after the audio of the Student Agent's response finishes. So one of the feedback points was to leave what the Student Agent said in the 3D environment to let the teacher re-read as much as they want. Another participant mentioned how they wished they could talk to the Student Agent outside of class so that the they could talk to them alone and help them calm down. The last comment mentioned is the inability to express certain tones and actions when talking to the Student Agents, which they felt caused the Student Agents to misunderstand the exact intentions of the participant.

5.2 | Results

In this section, we go into detail on the results detailed in Section 5.1, and why we achieved these results. We split this section into two subsections, the Validation results and the Usability results.

5.2.1 | Validation Results

This section discusses the results gathered for each component, and what those results say about each component's performance. More specifically, we look into whether the responses generated by Class Simulation meets the educational professionals' expectations by generating responses which leverage the fine-tuning data created from expert responses. For the Orchestrator and Analyser, we do this by comparing the metric validation (BLEU, ROUGE, METEOR and Cosine Similarity) results of both models to show which set of responses were more similar to the fine-tuning data, and by looking at which generated responses the experts preferred from the expert validation. Finally for the Student Agent, we compare the metric (`emotion-english-distilroberta-base` (Hartmann (2022)) model) and expert validation results with the original emotions used to generate the Student Agent responses to find out whether the emotions used were correctly reflected in the generated response.

5.2.1.1 | Orchestrator

Looking at the expert validation of the Orchestrator, the results show a varied performance with most experts picking the "both" option. These results show us that the fine-tuning didn't make enough of an impact that the Orchestrator would generate more realistic scenarios. This was also reflected in the metric evaluation in which most scores, although were slightly in favour of the Orchestrator, were still not different enough to say that the fine-tuning helped the Orchestrator generate more realistic scenarios.

5.2.1.2 | Student Agent

Looking at the results of the Student Agent, we can see that the results of the expert and metric validation almost match consistently with the original emotions used to generate the response. Comparing the expert and metric validation, responses one and three match completely, whilst response two does not match at all, with five participants ranking sadness the highest and one participant ranking neutral, and the emotion classifier giving the highest score to neutral. Comparing these with the emotions used originally, the joy emotion classified matches with the excited, happy and alert emotions used, whilst for response three, the anger emotion classified matches with stressed, nervous, and upset. For response two, sadness nor neutral match with the emotions stressed, nervous and upset. These results could signify that the difference between one emotion and another could be difficult to notice, especially through text. Other forms of communication could have helped the teacher confidently answer which emotions are prevalent in a Student Agent's response.

5.2.1.3 | Analyser

Similarly to the Orchestrator, the results of the Analyser show a varied performance with fine-tuning. For question one experts preferred both pieces of feedback equally, whilst for question two, the experts overwhelmingly preferred the feedback generated by the Analyser. The results from the metric evaluation show varied results, with the Analyser slightly outperforming the non-fine-tuned GPT-4o mini model in most cases, whilst underperforming in others. These results show that the fine-tuned Analyser did not generate feedback which is more similar to feedback received in real life as compared to the GPT-4o mini model.

5.2.2 | Usability Results

This section looks into the results gathered from the participants who used Class Simulation. The Usability results help us to understand whether Class Simulation provided pedagogically valuable and implementable feedback for teachers. More specifically, the results of the participant's Perceived Usefulness of Class Simulation gives us a good idea of what the participants thought of the generated responses. Along with the above, we calculate the scores from the SUS and GUESS questionnaires to get a score on how usable and engaging the participants found our system..

5.2.2.1 | Teacher Opinions

In this section, we look at the opinions of the participants who used Class Simulation.

Perceived Usefulness The participants agreed on all of the questions asked within this section. From this, the participants agree that Class Simulation can 1) help teachers improve handling of disruptive behaviours, 2) expose the teacher to issues which can occur during class, and 3) provide helpful feedback for teachers to use in real life. Since none of the participants required any further clarification on the feedback, we could not measure whether the Mentor helped provide further clarification on the analysis. This also shows that the feedback provided was clear and detailed enough that the participants understood the feedback fully, and didn't require clarification.

Perceived Compatibility The opinions of participants on the compatibility of Class Simulation is a bit more varied. All the participants agreed that the scenario and feedback generated were similar to real-life scenarios and feedback. When it came to the Student Agent's responses, one person disagreed that they generated responses similar to real-life students. This falls in line with the feedback from the usability experiments, in which participants mentioned how Student Agents were too apologetic, and easily recognised their emotions.

General Technology Proficiency The participants' technology proficiency varied. Whilst many participants reported feeling confident using technologies such as ChatGPT, educational training applications, and other digital learning tools, some participants indicated a neutral level of confidence. We attribute the variety of responses to the different back-

grounds of the participants, with some still pursuing their Masters in Education, and others having years of experience working in the educational sector.

5.2.2.2 | System Usability Scale

To calculate the score of the SUS questions, we go through the following steps:

1. Convert the answers into scores from 1 to 5, with Strongly Disagree being 1 and Strongly Agree being 5 (as seen in Table 5.7).
2. If the question is positively-oriented, then subtract 1 from the participant's score. If the question is negatively-oriented, then subtract the participant's score from 5 (as seen in Table 5.8).
3. Add all of the adjusted scores together for each participant, which will get us a value between 6 and 24 (6 responses multiplied by a maximum adjusted score of 4) (as seen in Table 5.9).
4. Finally, multiply the total score by a multiplier which scales the value between 0 and 100 (as seen in Table 5.9). In this case, the multiplier will be 100 division by 24 (maximum possible score) which gives us 4.17.
5. Repeat these steps for all participants, and then calculate the average score by adding up the total scaled scores of each participant, and dividing it by the number of participants (as seen in Equation 5.1).

$$\frac{91.67 + 75 + 70.83 + 95.83 + 70.83 + 87.50 + 95.83 + 79.17 + 62.50 + 75}{10} = 80.42 \tag{5.1}$$

Given our score of 80.42, we look at Sauro and Lewis (2016)'s curved grading scale (as seen in Table 5.10) to get an idea of how well our system performed. They used data from 241 industrial usability studies and surveys to create a curved grading scale with a SUS score of 68 at the centre of the range for a "C". With a score of 80.42, our system gets a grade of A-, falling under the 85th to 89th percentile.

	Question 1	Question 2	Question 3	Question 4	Question 5	Question 6
P1	5	2	4	1	1	5
P2	3	4	5	1	2	5
P3	4	3	4	2	1	3
P4	5	1	5	1	1	4
P5	4	4	4	1	3	5
P6	5	4	5	1	1	5
P7	5	1	5	1	1	4
P8	5	3	5	2	2	4
P9	5	4	3	2	3	4
P10	5	4	4	2	2	5

Table 5.7: Participant SUS responses converted into Values. Strongly Disagree = 1, Disagree = 2, Neutral = 3, Agree = 4 and Strongly Agree = 5. (Source: John Carmel Aquilina)

	Question 1 (Positive)	Question 2 (Negative)	Question 3 (Positive)	Question 4 (Negative)	Question 5 (Negative)	Question 6 (Positive)
P1	4	3	3	4	4	4
P2	2	1	4	4	3	4
P3	3	2	3	3	4	2
P4	4	4	4	4	4	3
P5	3	1	3	4	2	4
P6	4	1	4	4	4	4
P7	4	4	4	4	4	3
P8	4	2	4	3	3	3
P9	4	1	2	3	2	3
P10	4	1	3	3	3	4

Table 5.8: Participant response values adjusted based on question polarity. If question is positively-oriented, then we subtract 1 from the response value, and if question is negatively-oriented, we subtract the response value from 5. (Source: John Carmel Aquilina)

	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10
Total Score	22	18	17	23	17	21	23	19	15	18
Total Scaled Score	91.67	75	70.83	95.83	70.83	87.50	95.83	79.17	62.50	75

Table 5.9: Total Score calculated per participant from 6 to 24, and Total Scaled Score per participant from 1 to 100. The Maximum for the Total Score is calculated by multiplying the maximum score after adjustment and the number of questions within the questionnaire. The scores are scaled by first dividing 100 by the maximum possible score to get the scale, then multiplying the Total Score of each participant by the calculated scale. (Source: John Carmel Aquilina)

Grade	SUS	Percentile range
A+	84.1 – 100	96 – 100
A	80.8 – 84.0	90 – 95
A-	78.9 – 80.7	85 – 89
B+	77.2 – 78.8	80 – 84
B	74.1 – 77.1	70 – 79
B-	72.6 – 74.0	65 – 69
C+	71.1 – 72.5	60 – 64
C	65.0 – 71.0	41 – 59
C-	62.7 – 64.9	35 – 40
D	51.7 – 62.6	15 – 34
F	0 – 51.6	0 – 14

Table 5.10: Curved Grading Scale for the SUS. (Source: Sauro and Lewis (2016))

5.2.2.3 | Game User Experience Satisfaction Scale

To calculate the GUESS score, we follow a process similar to the SUS score calculation:

1. Convert the answers into scores from 1 to 5, with Strongly Disagree being 1 and Strongly Agree being 5 (as seen in Table 5.11).

2. Add all of the converted scores together for each participant, which will get us a value between 5 and 25 (5 responses multiplied by a maximum adjusted score of 5) (as seen in Table 5.12).

3. Finally, multiply the total score by a multiplier which scales the value between 0 and 100 (as seen in Table 5.12). In this case, the multiplier will be 100 division by 25 (maximum possible score) which gives us 4.

4. Repeat these steps for all participants, and then calculate the average score by adding up the total scaled scores of each participant, and dividing it by the number of participants (as seen in Equation 5.2).

$$\frac{84 + 76 + 84 + 84 + 88 + 100 + 100 + 80 + 76 + 96}{10} = 86.8 \tag{5.2}$$

A GUESS score of 86.8 shows us that Class Simulation and its components managed to engage the participants. The question with the least score was question 3, which asked whether the participants thought the characters in the simulation responded realistically.

	Question 1	Question 2	Question 3	Question 4	Question 5
P1	5	4	2	5	5
P2	4	4	4	3	4
P3	4	4	4	5	4
P4	3	4	4	5	5
P5	4	5	4	4	5
P6	5	5	5	5	5
P7	5	5	5	5	5
P8	4	4	4	4	4
P9	3	4	4	5	3
P10	5	5	4	5	5

Table 5.11: Participant GUESS responses converted into Values. Strongly Disagree = 1, Disagree = 2, Neutral = 3, Agree = 4 and Strongly Agree = 5. (Source: John Carmel Aquilina)

	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10
Total Score	21	19	21	21	22	25	25	20	19	24
Total Scaled Score	84	76	84	84	88	100	100	80	76	96

Table 5.12: Total Score calculated per participant from 5 to 25, and Total Scaled Score per participant from 1 to 100. The Maximum for the Total Score is calculated by multiplying the maximum score after adjustment and the number of questions within the questionnaire. The scores are scaled by first dividing 100 by the maximum possible score to get the scale, then multiplying the Total Score of each participant by the calculated scale. (Source: John Carmel Aquilina)

5.3 | Discussion of Findings

In this section, we revisit the research questions specified in Chapter 1 to see whether we reached them with the results achieved from the experiments. We also discuss several ethical considerations related to the research questions.

The first research question is defined as follows:

Does the class simulation system provide pedagogically valuable and implementable feedback that impacts teachers' classroom management strategies?

The results gathered from the Usability experiment help to answer this question. From the first part of the questionnaire, the participant's answers on the Perceived Usefulness show that they agreed that Class Simulation can 1) help teachers improve their handling of disruptive behaviours in class, 2) expose teachers to issues which can occur during class, and 3) provide helpful feedback for teachers to use in a real life class scenario. The participants could not give any feedback on the Mentor as none of them requested

for further clarification on the feedback. An almost similar result can be seen in the participant's Perceived Compatibility of Class Simulation. All participants agreed that the scenario generated by the Orchestrator and the feedback generated by the Analyser were similar to real-life classroom scenarios and feedback given by real-life professionals. When it comes to the realism of the responses generated by the Student Agent, all but one participant agreed that the responses are similar to real-life students responses. The second part of the questionnaire related to the Usability of Class Simulation. With a calculated SUS score of 80.42, most of the participants agreed that the system was relatively easy to use, did not require a lot of training, and could be useful to train teachers. Some participants noted that they would need the support of a technical person to use Class Simulation, showing the need for to "tutorial" to guide teachers better. The third part of the questionnaire concerns the Engagement of Class Simulation. A calculated GUESS score of 86.8 shows that the participants were generally engaged with the scenario generated from Class Simulation. Looking into the responses, the participants agreed that the scenario captivated them from the start and was clearly understandable, the Student Agent's characters were well developed, and there was an interest in seeing how events in the simulation will progress. Similarly to the first part of the questionnaire, one out of the ten participants thought that the characters in the simulation did not responded realistically.

The second research question is defined as follows:

To what extent does our implementation generate scenarios, student behaviours, and feedback that align with educational professionals' expectations?

During the development of Class Simulation, we collected data from real-life teachers and experts regarding potential real-life classroom scenarios and the best way they would handle it. This was then used to fine-tune the models used within the Orchestrator and Analyser. The models in the Student Agent were not fine-tuned due to a lack of student data. The results gathered from the Expert and Metric Validation experiment helps to decide whether this fine-tuning data had a clear effect on the model and the results generated, and to answer the second research question defined above. For the Orchestrator, the experts decided that both models produced scenarios which are likely to happen, with the exception of the last question in which experts preferred the scenario generated from the non-fine-tuned GPT-4o mini model. The calculated metrics (BLEU, ROUGE, METEOR

and Cosine Similarity) indicated that the Orchestrator's outputs were slightly more similar to the fine-tuning data compared to the non-fine-tuned GPT-4o mini model, however, this difference was not substantial enough to conclude that the fine-tuning had a huge effect on the results generated by the model. For the Student Agent, the experts mostly agreed on the emotions present in the responses generated. That being said, the emotions agreed upon by the experts only matched correctly for two out of the three responses generated by the Student Agent. Similarly for the metric evaluation, the emotions classified using Hartmann (2022)'s `emotion-english-distilroberta-base` model matched with the expert's assigned values for two out of three of the responses. The one response which did not have matching emotions between the experts and the emotion classifier is also the one which did not match with the original emotions used. For the Analyser, the experts were provided with two pairs of feedback. For the first pair, the experts equally preferred both pieces of feedback generated, whilst for the second one, the experts preferred the feedback generated by the Analyser, showing a slight preference to it. Similar to the Orchestrator, the Analyser's calculated metrics showed mixed results. In some metrics, the Analyser slightly outperformed the non-fine-tuned GPT-4o mini model, whilst also underperforming in other metrics. This also shows that there was not a difference substantial enough to conclude that the fine-tuning had an effect on the results generated by the Analyser. The results gathered related to the Orchestrator and Analyser can be mainly attributed to a lack of fine-tuning data, with typical fine-tuning data containing hundreds or even thousands of data points.

Related to these research questions, we look into several ethical concerns which need to be considered before deploying Class Simulation for real life use cases. For research question one, the main concern relates to the teacher and user privacy when using Class Simulation. This was easily ensured for the experiments due to the participants not entering any information and not utilising their personal devices for the experiments, but future implementations need to ensure that any data entered or collected is stored safe and securely. A safe-guard must also be implemented for the generated content to protect against biases and stereotypes, such as assigning a specific role to a student solely based on their ethnicity or gender. This safeguard would be implemented by artificially generating data to counteract any potential stereotypes and ensure balance in gender and ethnicity representation. A similar safe-guard needs to also be applied to the Analyser to ensure that feedback generated and presented to teachers cannot be misinterpreted, therefore potentially negatively effecting how teachers respond to real-life

students. For research question two, we must ensure that any datasets used to train and fine-tune models do not include any biases which could lead to generating an incorrect response, such as a scenario which favours the disruptive behaviour, or feedback which should not be implemented in real-life classrooms. Looking at the teacher and Student Agent communication, we need to ensure that 1) the sentiment analysis performed on the teacher's input is better at predicting the correct sentiment which the teacher is presenting, and 2) the emotions used to generate the Student Agent's response is represented as best as possible within the generated response and the Student Agent itself. Along with this, we also need to improve on the current emotion model to ensure a more accurate representation of student and teacher emotions.

5.4 | Conclusion

In this section, we looked into the data gathered from the experiments. First, we detail how we evaluated the Class Simulation system and present the data in a readable format. Then, we discuss the results gathered, going into detail on what the results mean. Finally, we related the results back to the research questions defined in Chapter 1, and discuss any ethical concerns related to the results and the research questions.

6

Conclusion & Future Work

In this section, we conclude the entire project by going through the following: First, we will revisit the aims and objectives defined within the introduction, deciding whether we have successfully reached them through our implementation. Second, we will detail the contribution of our work in the education and AI fields. Then, we will mention three critiques and limitations, noted by us and by participants throughout the experimentation phase; We will also go through the future work, which discuss several points of priority when implementing future versions of Class Simulation; and finally, we give our final remarks of the project, in which we close the project off with some last words.

6.1 | Revisiting Aims and Objectives

This research set out to explore how LLMs can be adapted and fine-tuned to simulate classroom disruptive behaviour scenarios for teacher training. We focus on answering two key questions; Whether the simulations generated using the fine-tuned LLM can provide pedagogically valuable feedback, and to what extent the LLMs can authentically model emotional and behavioural classroom dynamics.

The findings suggest that while Class Simulation demonstrated success as a POC, it also revealed the underperformance of the fine-tuning went into the Orchestrator and Analyser, showing a lack of substantial gains in performance. This shows us the technical and methodological challenges of fine-tuning general purpose LLMs using small domain-specific educational datasets. Usability testing confirmed that teachers perceived the simulation as engaging and pedagogically useful, however Validation testing showed only slight improvements following fine-tuning with the results of the calculated BLEU, ROUGE, METEOR and Cosine Similarity metrics indicating unsubstantial partial gains. For example: An average difference of 0.052 in BLEU score and an average difference of -0.029 in Cosine Similarity between the responses generated by the Orchestrator and a non-fine-tuned variant. These scores fell short of statistical significance. This points to limitations in data volume and diversity, as typical fine-tuning datasets in NLP research

would contain hundreds to thousands of samples, whilst our dataset contained less than fifty samples.

Affective modelling through the Student Agent was another critical aspect. Whilst the emotion classifier (based on the emotions used in Hartmann (2022)'s `emotion-english-distilroberta-base` model) was able to assign emotions to textual responses, its integration into the generative loop did not always yield behaviourally consistent outputs. This highlights the difficulty of transferring affective features into coherent multi-modal expressions, an open research problem in emotion aware AI systems. A similar issue occurred with emotion representation within the Student Agent. Even though we directed the Student Agent to incorporate the clearly incorporate its active emotions into the generated text, the experts and the emotion classifier still incorrectly predicted what emotions were present in the text, highlighting a clear need for multi-modal expressions.

Class Simulation also highlighted the importance of ethical AI implementations and the data limitation issues. With Class Simulation focusing on the generation of pedagogically valuable simulations, future versions should integrate open-weight LLMs (Such as Meta's Llama-3 and Mistral) to ensure transparency, fairness, and respect of user privacy. Data limitations during the fine-tuning of the LLMs significantly impacted our results. The small size of the fine-tuning data led to performance issues in our fine-tuned model when compared to the non-fine-tuned baseline. Additionally, inherent biases within the GPT-4o mini model (due to the diverse and unknown data sources used to train it) further influenced the outcomes. A critical challenge also occurs when ensuring reproducibility of the fine-tuning process. Variability in the processing of the fine-tuning data and fine-tuning settings can lead to inconsistent results. Therefore, it is required to document the creation of the fine-tuning data and all of the configurations used throughout the fine-tuning process to ensure reproducibility within the LLMs.

Overall, the study contributes empirical insight into model adaptation and domain generalisation limits when fine-tuning LLMs on small educational corpora. Whilst the objectives of the functional implementation were met, the AI integrations were only partially met. For future versions of Class Simulation to meet the AI objectives, we would need to scale up and diversify the educational data used, improve the affective modelling by integrating multi-modal emotion modelling, and utilise open-weight LLMs which are easily reproducible and contain minimal model biases.

6.2 | Contribution of Work

Our work contributes into two separate fields:

- **AI Methodological Contributions**: Our contribution can be separated into three:
  1. Fine-tuning pipeline for small educational corpora. Within this project, we create a pipeline for transforming expert scenario and feedback data into fine-tuning data for fine-tuning LLMs using DPO. Although the gains recorded in the Validation results were not substantial enough to show clear improvement, this still provides evidence on the lower threshold of domain transfer in LLM.
  2. Emotion modelling in generative dialogue. By integrating emotions into response generation, the system highlights the fragility of emotion and text alignment within current Transformer architectures.
  3. Evaluation methodology which bridges qualitative and quantitative measures. Combining the use of human expert validation and calculation of NLP metrics (BLEU, ROUGE, METEOR and Cosine Similarity) offers a hybrid framework for measuring the pedagogical usefulness and linguistic fidelity, contributing to evaluation practices in educational AI systems.
- **Educational Contributions**: Class Simulation demonstrates the feasibility of using multi-agent LLM systems as training tools for classroom management. Teachers valued the reflective dialogue and contextual feedback mechanisms, identify the tool as a low-risk environment for testing out different strategies for handling disruptive behaviours. The low-risk environment supports Class Simulation's potential as a complement to practical based training.

6.3 | Critique and Limitations

This section covers three main critiques in our POC noted by myself and several participants:

1. Small-scale FT Data and Limited Generalisation: From an AI research perspective, the main limitation lies in the scale and scope of the fine-tuning data. Both the Orchestrator and Analyser models were fine-tuned with fewer than fifty examples, constraining their capacity to generalise. The Student Agent model relied on zero-shot prompting which reduced emotional authenticity.

2. Model Dependence and Reproducibility Challenges: The use of OpenAI's GPT-4o mini model restricted transparency and reproducibility as internal fine-tuning parameters were inaccessible. This limits open benchmarking and replication, both essential to credible AI research.

3. Data Bias and Emotion Classification Fragility: From an ethics perspective, the data captured a very narrow demographic and linguistic range. This raises potential biases in generated behaviours and emotional inferences. Although safe-guards were conceptually planned (such as balanced genders and cultural variety), their implementation was incomplete due to data scarcity. This study also highlights methodological fragility in emotion modelling pipelines when sentiment classifiers misinterpret intent. This leads to responses being pedagogically inconsistent with the teachers' intent, showing the importance of interpretability in effective AI research.

6.4 | Future Work

Building on the insights gathered from our work, future research should prioritise deeper exploration of AI model behaviour, not just improved usability:

1. Model Adaptation and Generalisation: Datasets can be scaled up using data augmentation, in which pre-existing data is used to create new data samples, helpful in improving model optimisation and generalise-ability. We can also explore the use of adapter-based fine-tuning in which only a small percentage of the model's parameters are updated. Finally, we can also look into RLHF by rewarding the LLMs when a realistic scenario, a realistic student response, or useful feedback is generated.

2. Affective modelling: The integration of multimodal emotion recognition could generate more accurate emotions when the teacher is talking to the Student Agent. This could include using facial features and voice characteristics (tone, pitch, and cadence) to detect emotions. Emotion-action mapping could also be implemented, in which emotions are mapped to specific actions, such as if the Student Agent is angry, they can suddenly leave the room or throw an item.

3. Evaluation and Ethics: Explainable metrics could be integrated into Class Simulation to explain why certain feedback or emotional responses are generated, enabling

more transparency and trust. Further transparency can be applied by using open-weight models such as Meta's Llama-3 and Mistral instead of OpenAI's GPT-4o mini model. Using open-weight models makes it easier to access the internals for better understanding, and makes it more secure as data remains on-premises as compared to sending it to OpenAI's models to generate a response.

6.5 | Final Remarks

This research contributes an early but significant step towards understanding how LLMs can be adapted, emotionally aligned, and validated for domain specific use in education. This project demonstrates the technical feasibility and educational promise of LLM-driven simulations. Class Simulation also shows the challenges of accurate emotion recognition and realistic emotion modelling, emphasising the importance of interpretability when it comes to handling the human emotions which are inherently complex and subjective.

References

Achiam, Adler, S., Agarwal, S., Ahmad, L., Akkaya, I., Aleman, F. L., Almeida, D., Altenschmidt, J., Altman, S., Anadkat, S., et al. Gpt-4 technical report, 2024.

Ahn, M., Brohan, A., Brown, N., Chebotar, Y., Cortes, O., David, B., Finn, C., Fu, C., Gopalakrishnan, K., Hausman, K., et al. Do as i can, not as i say: Grounding language in robotic affordances, 2022.

Alhafni, B., Vajjala, S., Bannò, S., Maurya, K. K., and Kochmar, E. Llms in education: Novel perspectives, challenges, and opportunities, 2024. URL https://arxiv.org/abs/2409.11917.

Alva-Manchego, F., Martin, L., Bordes, A., Scarton, C., Sagot, B., and Specia, L. ASSET: A dataset for tuning and evaluation of sentence simplification models with multiple rewriting transformations. In Jurafsky, D., Chai, J., Schluter, N., and Tetreault, J., editors, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4668–4679, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.424. URL https://aclanthology.org/2020.acl-main.424/.

Bahdanau, D., Cho, K., and Bengio, Y. Neural machine translation by jointly learning to align and translate, 2016.

Banerjee, S. and Lavie, A. METEOR: An automatic metric for MT evaluation with improved correlation with human judgments. In Goldstein, J., Lavie, A., Lin, C.-Y., and Voss, C., editors, *Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*, pages 65–72, Ann Arbor, Michigan, June 2005. Association for Computational Linguistics. URL https://aclanthology.org/W05-0909/.

Bannò, S., Ma, R., Qian, M., Knill, K. M., and Gales, M. J. F. Towards end-to-end spoken grammatical error correction. In *ICASSP 2024 - 2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 10791–10795, 2024.

doi: 10.1109/ICASSP48485.2024.10446782.

Bassil, Y. A simulation model for the waterfall software development life cycle, 2012.

Binz, M. and Schulz, E. Using cognitive psychology to understand gpt-3. *Proceedings of the National Academy of Sciences*, 120(6):e2218523120, 2023.

Bourgin, D. D., Peterson, J. C., Reichman, D., Griffiths, T. L., and Russell, S. J. Cognitive model priors for predicting human decisions, 2019. URL https://arxiv.org/abs/1905.09397.

Bowman, S. R., Angeli, G., Potts, C., and Manning, C. D. A large annotated corpus for learning natural language inference, 2015. URL https://arxiv.org/abs/1508.05326.

Brooke, J. SUS-A quick and dirty usability scale. *Usability evaluation in industry*, 189 (194), 1996.

Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al. Language models are few-shot learners, 2020.

Bruner, J. S. *The process of education*. Harvard university press, 2009.

Bryant, C., Felice, M., and Briscoe, T. Automatic annotation and evaluation of error types for grammatical error correction. In Barzilay, R. and Kan, M.-Y., editors, *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 793–805, Vancouver, Canada, July 2017. Association for Computational Linguistics. doi: 10.18653/v1/P17-1074. URL https://aclanthology.org/P17-1074/.

Bryant, C., Felice, M., Andersen, Ø. E., and Briscoe, T. The BEA-2019 shared task on grammatical error correction. In Yannakoudakis, H., Kochmar, E., Leacock, C., Madnani, N., Pilán, I., and Zesch, T., editors, *Proceedings of the Fourteenth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 52–75, Florence, Italy, August 2019. Association for Computational Linguistics. doi: 10.18653/v1/W19-4406. URL https://aclanthology.org/W19-4406/.

Calhoun, S., Carletta, J., Brenier, J. M., Mayo, N., Jurafsky, D., Steedman, M., and Beaver, D. The nxt-format switchboard corpus: a rich resource for investigating the syntax, semantics, pragmatics and prosody of dialogue. *Language resources and*

evaluation, 44:387–419, 2010.

Chiang, W.-L., Li, Z., Lin, Z., Sheng, Y., Wu, Z., Zhang, H., Zheng, L., Zhuang, S., Zhuang, Y., Gonzalez, J. E., et al. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality, March 2023. URL `https://lmsys.org/blog/2023-03-30-vicuna/`.

Chiu, T. K., Xia, Q., Zhou, X., Chai, C. S., and Cheng, M. Systematic literature review on opportunities, challenges, and future research recommendations of artificial intelligence in education. *Computers and Education: Artificial Intelligence*, 4:100118, 2023. ISSN 2666-920X. doi: https://doi.org/10.1016/j.caeai.2022.100118.

Cho, K., van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., and Bengio, Y. Learning phrase representations using RNN encoder–decoder for statistical machine translation. In Moschitti, A., Pang, B., and Daelemans, W., editors, *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734, Doha, Qatar, October 2014. Association for Computational Linguistics. doi: 10.3115/v1/D14-1179. URL `https://aclanthology.org/D14-1179`.

Cobbe, K., Kosaraju, V., Bavarian, M., Chen, M., Jun, H., Kaiser, L., Plappert, M., Tworek, J., Hilton, J., Nakano, R., et al. Training verifiers to solve math word problems, 2021.

Coleman, M. and Liau, T. L. A computer readability formula designed for machine scoring. *Journal of Applied Psychology*, 60(2):283, 1975.

Cooper, J. T., Gage, N. A., Alter, P. J., LaPolla, S., MacSuga-Gage, A. S., and Scott, T. M. Educators' self-reported training, use, and perceived effectiveness of evidence-based classroom management practices. *Preventing School Failure: Alternative Education for Children and Youth*, 62(1):13–24, 2018.

Council of Europe. *Common European Framework of Reference for Languages: Learning, Teaching, Assessment*. Cambridge University Press, Cambridge, 2001.

Dahlmeier, D., Ng, H. T., and Wu, S. M. Building a large annotated corpus of learner English: The NUS corpus of learner English. In Tetreault, J., Burstein, J., and Leacock, C., editors, *Proceedings of the Eighth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 22–31, Atlanta, Georgia, June 2013. Association for

Computational Linguistics. URL https://aclanthology.org/W13-1703/.

de Vicente Mohino, J., Bermejo Higuera, J., Bermejo Higuera, J. R., and Sicilia Montalvo, J. A. The application of a new secure software development life cycle (s-sdlc) with agile methodologies. *Electronics*, 8(11), 2019. ISSN 2079-9292. doi: 10.3390/electronics8111218.

Demszky, D., Movshovitz-Attias, D., Ko, J., Cowen, A., Nemade, G., and Ravi, S. GoEmotions: A dataset of fine-grained emotions. In Jurafsky, D., Chai, J., Schluter, N., and Tetreault, J., editors, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4040–4054, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.372. URL https://aclanthology.org/2020.acl-main.372/.

Dolan, W. B. and Brockett, C. Automatically constructing a corpus of sentential paraphrases. In *Proceedings of the Third International Workshop on Paraphrasing (IWP2005)*, 2005. URL https://aclanthology.org/I05-5002/.

Dua, D., Wang, Y., Dasigi, P., Stanovsky, G., Singh, S., and Gardner, M. DROP: A reading comprehension benchmark requiring discrete reasoning over paragraphs. In Burstein, J., Doran, C., and Solorio, T., editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2368–2378, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1246. URL https://aclanthology.org/N19-1246.

Felice, M., Bryant, C., and Briscoe, T. Automatic extraction of learner errors in ESL sentences using linguistically enhanced alignments. In Matsumoto, Y. and Prasad, R., editors, *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 825–835, Osaka, Japan, December 2016. The COLING 2016 Organizing Committee. URL https://aclanthology.org/C16-1079/.

Geertzen, J., Alexopoulou, T., Korhonen, A., et al. Automatic linguistic annotation of large scale l2 databases: The ef-cambridge open language database (efcamdat). In *Proceedings of the 31st Second Language Research Forum. Somerville, MA: Cascadilla Proceedings Project*, pages 240–254, 2013.

Gehman, S., Gururangan, S., Sap, M., Choi, Y., and Smith, N. A. Realtoxicityprompts:

Evaluating neural toxic degeneration in language models, 2020. URL https://arxiv.org/abs/2009.11462.

Gehring, J., Auli, M., Grangier, D., Yarats, D., and Dauphin, Y. N. Convolutional sequence to sequence learning, 2017.

Gettinger, M., Stoiber, K., and Koscik, R. Effects of a preparation program focused on accommodating children with challenging behaviors. *Teacher Education and Special Education*, 31(3):164–181, 2008. doi: 10.1177/0888406408330624.

Geva, M., Khashabi, D., Segal, E., Khot, T., Roth, D., and Berant, J. Did aristotle use a laptop? a question answering benchmark with implicit reasoning strategies, 2021.

Ghosh, A., Heffernan, N., and Lan, A. S. Context-aware attentive knowledge tracing, 2020. URL https://arxiv.org/abs/2007.12324.

Gunning, R. The technique of clear writing. 1952.

Guo, X., Huang, Z., Gao, J., Shang, M., Shu, M., and Sun, J. Enhancing knowledge tracing via adversarial training, 2021. URL https://arxiv.org/abs/2108.04430.

Gururangan, S., Swayamdipta, S., Levy, O., Schwartz, R., Bowman, S., and Smith, N. A. Annotation artifacts in natural language inference data. In Walker, M., Ji, H., and Stent, A., editors, *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 107–112, New Orleans, Louisiana, June 2018. Association for Computational Linguistics. doi: 10.18653/v1/N18-2017. URL https://aclanthology.org/N18-2017.

Hartmann, J. Emotion english distilroberta-base. https://huggingface.co/j-hartmann/emotion-english-distilroberta-base/, 2022.

Heemstra, F. Software cost estimation. *Information and Software Technology*, 34 (10):627–639, 1992. ISSN 0950-5849. doi: https://doi.org/10.1016/0950-5849(92)90068-Z.

Herbrich, R., Minka, T., and Graepel, T. Trueskill™: a bayesian skill rating system. *Advances in neural information processing systems*, 19, 2006.

Horton, J. J. Large language models as simulated economic agents: What can we learn

from homo silicus? Technical report, National Bureau of Economic Research, 2023.

Hosseini, M. J., Hajishirzi, H., Etzioni, O., and Kushman, N. Learning to solve arithmetic word problems with verb categorization. In Moschitti, A., Pang, B., and Daelemans, W., editors, *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 523–533, Doha, Qatar, October 2014. Association for Computational Linguistics. doi: 10.3115/v1/D14-1058. URL https://aclanthology.org/D14-1058.

Hu, B., Zhu, J., Pei, Y., and Gu, X. Exploring the potential of llm to enhance teaching plans through teaching simulation. *npj Science of Learning*, 10(1):7, 2025.

Jin, H., Lee, S., Shin, H., and Kim, J. Teach ai how to code: Using large language models as teachable agents for programming education. In *Proceedings of the 2024 CHI Conference on Human Factors in Computing Systems*, pages 1–28, 2024.

Jin, H., Yoo, M., Park, J., Lee, Y., Wang, X., and Kim, J. Teachtune: Reviewing pedagogical agents against diverse student profiles with simulated students, 2025. URL https://arxiv.org/abs/2410.04078.

Jørgensen, M. and Shepperd, M. A systematic review of software development cost estimation studies. *Software Engineering, IEEE Transactions on*, 33:33–53, 02 2007. doi: 10.1109/TSE.2007.256943.

Kaneko, M. and Okazaki, N. Controlled generation with prompt insertion for natural language explanations in grammatical error correction. In Calzolari, N., Kan, M.-Y., Hoste, V., Lenci, A., Sakti, S., and Xue, N., editors, *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 3955–3961, Torino, Italia, May 2024. ELRA and ICCL. URL https://aclanthology.org/2024.lrec-main.350/.

Khan, R. A., Khan, S. U., Khan, H. U., and Ilyas, M. Systematic literature review on security risks and its practices in secure software development. *ieee Access*, 10: 5456–5481, 2022.

Khot, T., Clark, P., Guerquin, M., Jansen, P., and Sabharwal, A. Qasc: A dataset for question answering via sentence composition, 2020.

Kincaid, J. P., Fishburne Jr, R. P., Rogers, R. L., and Chissom, B. S. Derivation of new readability formulas (automated readability index, fog count and flesch reading ease

formula) for navy enlisted personnel. 1975.

Kolen, J. F. and Kremer, S. C. *Gradient Flow in Recurrent Nets: The Difficulty of Learning LongTerm Dependencies*, pages 237–243. 2001. doi: 10.1109/9780470544037.ch14.

Kriek, J. and Stols, G. Teachers' beliefs and their intention to use interactive simulations in their classrooms. *South African Journal of Education*, 30:439–456, 08 2010. doi: 10.15700/saje.v30n3a284.

Kruskal, W. H. and Wallis, W. A. Use of ranks in one-criterion variance analysis. *Journal of the American Statistical Association*, 47(260):583–621, 1952. doi: 10.1080/01621459.1952.10483441.

Laban, P., Schnabel, T., Bennett, P. N., and Hearst, M. A. SummaC: Re-visiting NLI-based models for inconsistency detection in summarization. *Transactions of the Association for Computational Linguistics*, 10:163–177, 2022. doi: 10.1162/tacl_a_00453.

Lakatos, R., Pollner, P., Hajdu, A., and Joó, T. Investigating the performance of retrieval-augmented generation and domain-specific fine-tuning for the development of ai-driven knowledge-based systems. *Machine Learning and Knowledge Extraction*, 7(1):15, February 2025. ISSN 2504-4990. doi: 10.3390/make7010015.

Lewis, M., Liu, Y., Goyal, N., Ghazvininejad, M., Mohamed, A., Levy, O., Stoyanov, V., and Zettlemoyer, L. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension, 2019. URL `https://arxiv.org/abs/1910.13461`.

Li, G., Hammoud, H. A. A. K., Itani, H., Khizbullin, D., and Ghanem, B. Camel: Communicative agents for "mind" exploration of large scale language model society, 2023.

Lin, B. Y., Lee, S., Khanna, R., and Ren, X. Birds have four legs?! NumerSense: Probing Numerical Commonsense Knowledge of Pre-Trained Language Models. In Webber, B., Cohn, T., He, Y., and Liu, Y., editors, *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6862–6868, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.557. URL `https://aclanthology.org/2020.emnlp-main.557`.

Lin, C.-Y. ROUGE: A package for automatic evaluation of summaries. In *Text Sum-*

marization Branches Out, pages 74–81, Barcelona, Spain, July 2004. Association for Computational Linguistics. URL https://aclanthology.org/W04-1013/.

Lin, S., Hilton, J., and Evans, O. Truthfulqa: Measuring how models mimic human falsehoods, 2022. URL https://arxiv.org/abs/2109.07958.

Liu, J., Liu, A., Lu, X., Welleck, S., West, P., Le Bras, R., Choi, Y., and Hajishirzi, H. Generated knowledge prompting for commonsense reasoning. In Muresan, S., Nakov, P., and Villavicencio, A., editors, *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3154–3169, Dublin, Ireland, May 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.acl-long.225. URL https://aclanthology.org/2022.acl-long.225.

Liu, Z., Liu, Q., Chen, J., Huang, S., and Luo, W. simplekt: A simple but tough-to-beat baseline for knowledge tracing, 2023. URL https://arxiv.org/abs/2302.06881.

Ludlow, K. *Official Quick Guide to Linguaskill*. Cambridge University Press & Assessment, Cambridge, 2020.

Marcus, M., Kim, G., Marcinkiewicz, M. A., MacIntyre, R., Bies, A., Ferguson, M., Katz, K., and Schasberger, B. The Penn Treebank: Annotating predicate argument structure. In *Human Language Technology: Proceedings of a Workshop held at Plainsboro, New Jersey, March 8-11, 1994*, 1994. URL https://aclanthology.org/H94-1020.

McCoy, R. T., Pavlick, E., and Linzen, T. Right for the wrong reasons: Diagnosing syntactic heuristics in natural language inference, 2019.

McGuire, S. N., Meadan, H., and Folkerts, R. Classroom and behavior management training needs and perceptions: A systematic review of the literature. 53(1):117–139. ISSN 1573-3319. doi: 10.1007/s10566-023-09750-z.

Mireles-Rios, R., Becchio, J. A., and Roshandel, S. Teacher evaluations and contextualized self-efficacy: Classroom management, instructional strategies and student engagement. *Journal of School Administration Research and Development*, 4(1):6–17, 2019.

Mohammad, S. and Bravo-Marquez, F. Emotion intensities in tweets. In Ide, N., Herbelot,

A., and Màrquez, L., editors, *Proceedings of the 6th Joint Conference on Lexical and Computational Semantics (*SEM 2017)*, pages 65–77, Vancouver, Canada, August 2017. Association for Computational Linguistics. doi: 10.18653/v1/S17-1007. URL https://aclanthology.org/S17-1007/.

Mohammad, S., Bravo-Marquez, F., Salameh, M., and Kiritchenko, S. Semeval-2018 task 1: Affect in tweets. In *Proceedings of the 12th international workshop on semantic evaluation*, pages 1–17, 2018.

Naismith, B., Mulcaire, P., and Burstein, J. Automated evaluation of written discourse coherence using GPT-4. In Kochmar, E., Burstein, J., Horbach, A., Laarmann-Quante, R., Madnani, N., Tack, A., Yaneva, V., Yuan, Z., and Zesch, T., editors, *Proceedings of the 18th Workshop on Innovative Use of NLP for Building Educational Applications (BEA 2023)*, pages 394–403, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.bea-1.32. URL https://aclanthology.org/2023.bea-1.32/.

Nangia, N., Vania, C., Bhalerao, R., and Bowman, S. R. CrowS-pairs: A challenge dataset for measuring social biases in masked language models. In Webber, B., Cohn, T., He, Y., and Liu, Y., editors, *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1953–1967, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.154. URL https://aclanthology.org/2020.emnlp-main.154/.

Ng, H. T., Wu, S. M., Wu, Y., Hadiwinoto, C., and Tetreault, J. The CoNLL-2013 shared task on grammatical error correction. In Ng, H. T., Tetreault, J., Wu, S. M., Wu, Y., and Hadiwinoto, C., editors, *Proceedings of the Seventeenth Conference on Computational Natural Language Learning: Shared Task*, pages 1–12, Sofia, Bulgaria, August 2013. Association for Computational Linguistics. URL https://aclanthology.org/W13-3601/.

Ng, H. T., Wu, S. M., Briscoe, T., Hadiwinoto, C., Susanto, R. H., and Bryant, C. The CoNLL-2014 shared task on grammatical error correction. In Ng, H. T., Wu, S. M., Briscoe, T., Hadiwinoto, C., Susanto, R. H., and Bryant, C., editors, *Proceedings of the Eighteenth Conference on Computational Natural Language Learning: Shared Task*, pages 1–14, Baltimore, Maryland, June 2014. Association for Computational Linguistics. doi: 10.3115/v1/W14-1701. URL https://aclanthology.org/

W14-1701/.

Niven, T. and Kao, H.-Y. Probing neural network comprehension of natural language arguments, July 2019. URL https://aclanthology.org/P19-1459/.

OpenAI. Introducing gpt-4.5, 2025. URL https://openai.com/index/introducing-gpt-4-5/. Accessed: 2025-05-03.

Ouyang, L., Wu, J., Jiang, X., Almeida, D., Wainwright, C. L., Mishkin, P., Zhang, C., Agarwal, S., Slama, K., Ray, A., et al. Training language models to follow instructions with human feedback, 2022. URL https://arxiv.org/abs/2203.02155.

Owais, M. and Ramakishore, R. Effort, duration and cost estimation in agile software development. pages 1–5, 08 2016. doi: 10.1109/IC3.2016.7880216.

Paperno, D., Kruszewski, G., Lazaridou, A., Pham, N. Q., Bernardi, R., Pezzelle, S., Baroni, M., Boleda, G., and Fernández, R. The LAMBADA dataset: Word prediction requiring a broad discourse context. In Erk, K. and Smith, N. A., editors, *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1525–1534, Berlin, Germany, August 2016. Association for Computational Linguistics. doi: 10.18653/v1/P16-1144. URL https://aclanthology.org/P16-1144.

Papineni, K., Roukos, S., Ward, T., and Zhu, W.-J. Bleu: a method for automatic evaluation of machine translation. In Isabelle, P., Charniak, E., and Lin, D., editors, *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA, July 2002. Association for Computational Linguistics. doi: 10.3115/1073083.1073135. URL https://aclanthology.org/P02-1040/.

Park, J. S., Popowski, L., Cai, C. J., Morris, M. R., Liang, P., and Bernstein, M. S. Social simulacra: Creating populated prototypes for social computing systems. 2022. URL https://arxiv.org/abs/2208.04024.

Park, J. S., O'Brien, J. C., Cai, C. J., Morris, M. R., Liang, P., and Bernstein, M. S. Generative agents: Interactive simulacra of human behavior, 2023. URL https://arxiv.org/abs/2304.03442.

Patel, A., Bhattamishra, S., and Goyal, N. Are NLP models really able to solve simple math word problems? In Toutanova, K., Rumshisky, A., Zettlemoyer, L., Hakkani-

Tur, D., Beltagy, I., Bethard, S., Cotterell, R., Chakraborty, T., and Zhou, Y., editors, *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2080–2094, Online, June 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.naacl-main.168. URL https://aclanthology.org/2021.naacl-main.168.

Peters, M. E. and Lecocq, D. Content extraction using diverse feature sets. In *Proceedings of the 22nd international conference on world wide web*, pages 89–90, 2013.

Phan, M. H., Keebler, J. R., and Chaparro, B. S. The development and validation of the game user experience satisfaction scale (guess). *Human Factors*, 58(8):1217–1247, 2016. doi: 10.1177/0018720816669646. PMID: 27647156.

Piaget, J. *The moral judgment of the child*. Routledge, 2013.

Piech, C., Spencer, J., Huang, J., Ganguli, S., Sahami, M., Guibas, L., and Sohl-Dickstein, J. Deep knowledge tracing, 2015. URL https://arxiv.org/abs/1506.05908.

Poria, S., Hazarika, D., Majumder, N., Naik, G., Cambria, E., and Mihalcea, R. MELD: A multimodal multi-party dataset for emotion recognition in conversations. In Korhonen, A., Traum, D., and Màrquez, L., editors, *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 527–536, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1050. URL https://aclanthology.org/P19-1050/.

Qian, C., Liu, W., Liu, H., Chen, N., Dang, Y., Li, J., Yang, C., Chen, W., Su, Y., Cong, X., et al. Chatdev: Communicative agents for software development, 2024. URL https://arxiv.org/abs/2307.07924.

Radford, A. and Narasimhan, K. Improving language understanding by generative pre-training. 2018. URL https://api.semanticscholar.org/CorpusID:49313245.

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., and Sutskever, I. Language models are unsupervised multitask learners. 2019.

Rae, J. W., Borgeaud, S., Cai, T., Millican, K., Hoffmann, J., Song, H. F., Aslanides, J., Henderson, S., Ring, R., Young, S., et al. Scaling language models: Methods, analysis

& insights from training gopher. *CoRR*, abs/2112.11446, 2021.

Raheja, V., Kumar, D., Koo, R., and Kang, D. CoEdIT: Text editing by task-specific instruction tuning. In Bouamor, H., Pino, J., and Bali, K., editors, *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 5274–5291, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.findings-emnlp.350. URL https://aclanthology.org/2023.findings-emnlp.350/.

Saravia, E., Liu, H.-C. T., Huang, Y.-H., Wu, J., and Chen, Y.-S. CARER: Contextualized affect representations for emotion recognition. In Riloff, E., Chiang, D., Hockenmaier, J., and Tsujii, J., editors, *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3687–3697, Brussels, Belgium, October-November 2018. Association for Computational Linguistics. doi: 10.18653/v1/D18-1404. URL https://aclanthology.org/D18-1404/.

Sauro, J. and Lewis, J. R. *Quantifying the user experience: Practical statistics for user research*. Morgan Kaufmann, 2016.

Scherer, K. R. and Wallbott, H. G. Evidence for universality and cultural variation of differential emotion response patterning., 1994.

Shank, M. K. and Santiague, L. Classroom management needs of novice teachers. *The Clearing house: a Journal of eduCaTional sTraTegies, issues and ideas*, 95(1):26–34, 2022.

Shao, Y., Li, L., Dai, J., and Qiu, X. Character-llm: A trainable agent for role-playing, 2023. URL https://arxiv.org/abs/2310.10158.

Shi, Y., Liang, R., and Xu, Y. Educationq: Evaluating llms' teaching capabilities through multi-agent dialogue framework, 2025. URL https://arxiv.org/abs/2504.14928.

Smith, E. A. and Senter, R. *Automated readability index*, volume 66. Aerospace Medical Research Laboratories, Aerospace Medical Division, Air . . . , 1967.

Stevenson, N. A., VanLone, J., and Barber, B. R. A commentary on the misalignment of teacher education and the need for classroom behavior management skills. 43(4): 393–404. ISSN 1934-8924. doi: 10.1007/s43494-020-00031-1.

Sutskever, I., Vinyals, O., and Le, Q. V. Sequence to sequence learning with neural

networks, 2014. URL https://arxiv.org/abs/1409.3215.

Talmor, A., Herzig, J., Lourie, N., and Berant, J. CommonsenseQA: A question answering challenge targeting commonsense knowledge. In Burstein, J., Doran, C., and Solorio, T., editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4149–4158, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1421. URL https://aclanthology.org/N19-1421.

Talmor, A., Yoran, O., Bras, R. L., Bhagavatula, C., Goldberg, Y., Choi, Y., and Berant, J. Commonsenseqa 2.0: Exposing the limits of ai through gamification, 2022. URL https://arxiv.org/abs/2201.05320.

Taori, R., Gulrajani, I., Zhang, T., Dubois, Y., Li, X., Guestrin, C., Liang, P., and Hashimoto, T. B. Stanford alpaca: An instruction-following llama model. https://github.com/tatsu-lab/stanford_alpaca, 2023.

Tran, H., Yao, Z., Li, L., and Yu, H. Readctrl: Personalizing text generation with readability-controlled instruction learning, 2024. URL https://arxiv.org/abs/2406.09205.

Upton, G. and Cook, I. *A Dictionary of Statistics 3e*. Oxford Paperback Reference. OUP Oxford, 2014. ISBN 9780199679188. URL https://books.google.com.mt/books?id=4WygAwAAQBAJ.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. Attention is all you need. *CoRR*, abs/1706.03762, 2017.

VYGOTSKY, L. S. *Mind in Society: Development of Higher Psychological Processes*. Harvard University Press, 1978. ISBN 9780674576285. URL http://www.jstor.org/stable/j.ctvjf9vz4.

Wang, A., Pruksachatkun, Y., Nangia, N., Singh, A., Michael, J., Hill, F., Levy, O., and Bowman, S. R. Superglue: A stickier benchmark for general-purpose language understanding systems, 2020. URL https://arxiv.org/abs/1905.00537.

Wang, X., Wei, J., Schuurmans, D., Le, Q., Chi, E., Narang, S., Chowdhery, A., and Zhou, D. Self-consistency improves chain of thought reasoning in language models, 2023. URL https://arxiv.org/abs/2203.11171.

Wei, J., Wang, X., Schuurmans, D., Bosma, M., Chi, E. H., Le, Q., and Zhou, D. Chain of thought prompting elicits reasoning in large language models. *CoRR*, abs/2201.11903, 2022.

Wei, J., Karina, N., Chung, H. W., Jiao, Y. J., Papay, S., Glaese, A., Schulman, J., and Fedus, W. Measuring short-form factuality in large language models, 2024. URL https://arxiv.org/abs/2411.04368.

Williams, A., Nangia, N., and Bowman, S. R. A broad-coverage challenge corpus for sentence understanding through inference, 2018. URL https://arxiv.org/abs/1704.05426.

Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., Cistac, P., Rault, T., Louf, R., Funtowicz, M., et al. Transformers: State-of-the-art natural language processing. In Liu, Q. and Schlangen, D., editors, *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online, October 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-demos.6. URL https://aclanthology.org/2020.emnlp-demos.6/.

Xu, S., Hu, D., Wang, R., and Zhang, X. Peer attention enhances student learning, 2023. URL https://arxiv.org/abs/2312.02358.

Xu, S., Zhang, X., and Qin, L. Eduagent: Generative student agents in learning, 2024. URL https://arxiv.org/abs/2404.07963.

Xu, S., Wen, H.-N., Pan, H., Dominguez, D., Hu, D., and Zhang, X. Classroom simulacra: Building contextual student generative agents in online education for learning behavioral simulation. In *Proceedings of the 2025 CHI Conference on Human Factors in Computing Systems*, CHI '25, page 1–26. ACM, April 2025. doi: 10.1145/3706598.3713773. URL http://dx.doi.org/10.1145/3706598.3713773.

Xu, W., Napoles, C., Pavlick, E., Chen, Q., and Callison-Burch, C. Optimizing statistical machine translation for text simplification. *Transactions of the Association for Computational Linguistics*, 4:401–415, 2016. doi: 10.1162/tacl_a_00107.

Zhang, J., Shi, X., King, I., and Yeung, D.-Y. Dynamic key-value memory networks for knowledge tracing, 2017. URL https://arxiv.org/abs/1611.08108.

Zhang, T., Kishore, V., Wu, F., Weinberger, K. Q., and Artzi, Y. Bertscore: Evaluating

text generation with bert, 2020. URL https://arxiv.org/abs/1904.09675.

Zhang, Y., Baldridge, J., and He, L. Paws: Paraphrase adversaries from word scrambling, 2019. URL https://arxiv.org/abs/1904.01130.

Zhong, M., Liu, Y., Yin, D., Mao, Y., Jiao, Y., Liu, P., Zhu, C., Ji, H., and Han, J. Towards a unified multi-dimensional evaluator for text generation, 2022. URL https://arxiv.org/abs/2210.07197.

Zhou, D., Schärli, N., Hou, L., Wei, J., Scales, N., Wang, X., Schuurmans, D., Cui, C., Bousquet, O., Le, Q., and Chi, E. Least-to-most prompting enables complex reasoning in large language models, 2023. URL https://arxiv.org/abs/2205.10625.

Zhu, Z., Bernhard, D., and Gurevych, I. A monolingual tree-based translation model for sentence simplification. In Huang, C.-R. and Jurafsky, D., editors, *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)*, pages 1353–1361, Beijing, China, August 2010. Coling 2010 Organizing Committee. URL https://aclanthology.org/C10-1152/.

A

Experiment Forms and Questionnaires

This appendix contains the forms submitted to University Research Ethics Committee (UREC), the consent form used within the Usability experiments, and the questionnaires provided to the teachers and experts.

A.1 | UREC Application Forms

L-Università ta' Malta

Research Ethics and Data Protection Form

University of Malta staff, students, or anyone else planning to carry out research under the auspices of the University, must complete this form. The UM may also consider requests for ethics and data protection review by External Applicants.

Ahead of completing this online form, please read carefully the University of Malta Research Code of Practice and the University of Malta Research Ethics Review Procedures. Any breach of the Research Code of Practice or untruthful replies in this form will be considered a serious disciplinary matter. It is advisable to download a full digital version of the form to familiarise yourself with its contents (https://www.um.edu.mt/research/ethics/resources/umdocuments/). You are also advised to refer to the FAQs (https://www.um.edu.mt/research/ethics/faqs).

Part 1: Applicant and Project Details

Applicant Details

Name: John Carmel

Surname: Aquilina

Email: john.c.aquilina.22@um.edu.mt

Applicant Status: Student

Please indicate if you form part of a Faculty, Institute, School or Centre: * Faculty of Information & Communication Technology

Department: * Department of Artificial Intelligence

Principal Supervisor's Name: * Vanessa Camilleri

Principal Supervisor's Email: * vanessa.camilleri@um.edu.mt

Co-Supervisor's Name:

Study Unit Code: * ICS5200

Course Title: * Master of Science in Artificial Intelligence [Taught and Research (Mainly by Research)]

Student Number: * 6803H

Project Details

Title of Research Project: *

Enhancing Teacher skills in handling Students Emotions using Large Language Models for the Generation of Classroom Scenarios

Project description, including research question/statement and method, in brief: *

Project Description:
This project aims to develop an AI-powered tool to assist teachers in enhancing their classroom management skills. By simulating realistic classroom scenarios featuring challenging student behaviours, the tool will provide teachers with personalised feedback and insights into effective strategies for managing these situations.

Research Statement:
This research investigates the potential of large language models (LLMs) to generate realistic classroom scenarios and analyse teacher responses, thereby offering a safe and supportive environment for teachers to practice and refine their classroom management techniques. The project prioritises anonymity and data privacy, ensuring that all student behavior data utilised is non-identifiable and generalised.

Methodology:
Scenario Generation: A fine-tuned LLM will be utilised to generate diverse classroom scenarios involving challenging student behaviours based on input from experienced teachers.
Student Agent Simulation: Another LLM-based agent will simulate student behavior within the scenarios, reacting to teacher actions and exhibiting realistic emotional responses.
Teacher Response Analysis: A specialised LLM model will analyse teacher responses to the scenarios, providing feedback based on best practices and expert knowledge.
Data Collection and Evaluation: Anonymous questionnaires will be used to gather data from teachers and classroom management experts, ensuring the accuracy and relevance of the generated scenarios and feedback.

Will project involve collection of primary data from human participants? Yes / Unsure

Explain primary data collection from human participants:

a. Salient participant characteristics (e.g. min-max participants, age, sex, other): *
5-10 participants who are all teachers with more than 2 years teaching experience at either primary or secondary levels of education in Malta;
Ideally there is an equal gender balance amongst the participants;

b. How will they be recruited (e.g. sampled, selected, contacted, etc.): *
Participants will be recruited over email or social media. Their participation is entirely voluntary.

c. What they will be required to do and for how long: *
They will be required to fill-in an anonymous Google Form in which they will answers come up with possible class scenarios and teacher actions. The form will take a maximum of 30 minutes to complete.

d. If inducements/rewards/compensation are offered: *
No rewards will be offered.

e. How participants/society may benefit: *

By combining the expertise of teachers and the power of AI, this project aims to create an innovative tool that empowers educators to create positive and supportive learning environments for all students.

f. Is the participant's identity recorded at any stage of the research (e.g. in consent forms, records, publications): *

No.

g. The manner in which you will manage and store the data and records (including consent forms): *

Data relevant to classroom situations, scenarios and behaviours will be stored locally and used to fine-tune LLMs.

Will project involve collection of primary data from animals (live non-human vertebrates/cephalopods, their foetuses and larvae, their tissue/samples and/or dead specimens of these animals)?

No

Part 2: Self Assessment and Relevant Details

Human Participants

1. Risk of harm to participants: No / N.A.

2. Physical intervention: No / N.A.

3. Vulnerable participants: No / N.A.

4. Identifiable participants: No / N.A.

5. Special Categories of Personal Data (SCPD): No / N.A.

6. Human tissue/samples: No / N.A.

7. Withheld info assent/consent: No / N.A.

8. 'opt-out' recruitment: No / N.A.

9. Deception in data generation: No / N.A.

10. Incidental findings: No / N.A.

Unpublished secondary data

11. Human: No / N.A.

12. Animal: No / N.A.

13. No written permission: No / N.A.

Animals

14. Live animals, lasting harm: No / N.A.

15. Live animals, harm: No / N.A.

16. Source of dead animals, illegal: No / N.A.

General Considerations

17. Cooperating institution: No / N.A.

18. Risk to researcher/s: No / N.A.

19. Risk to environment: No / N.A.

20. Commercial sensitivity: No / N.A.

Other Potential Risks

21. Other potential risks: No / N.A.

22. Official statement: Do you require an official statement from the F/REC that this submission has abided by the UM's REDP procedures?

No / N.A.

Part 3: Submission

Which F/REC are you submitting to? * Faculty of Information & Communication Technology

Attachments:

- Data_Collection_Tools.txt (size: 101 bytes, uploaded on: 29/07/2024 10:16:19)

- [ ] Information and/or recruitment letter*
- [x] Consent forms (adult participants)*
- [ ] Consent forms for legally responsible parents/guardians, in case of minors and/or adults unable to give consent*
- [ ] Assent forms in case of minors and/or adults unable to give consent*
- [x] Data collection tools (interview questions, questionnaire etc.)
- [ ] Data Management Plan
- [ ] Data controller permission in case of use of unpublished secondary data
- [ ] Licence/permission to use research tools (e.g. constructs/tests)
- [ ] Any permits required for import or export of materials or data
- [ ] Letter granting institutional approval for access to participants
- [ ] Institutional approval for access to data
- [ ] Letter granting institutional approval from person directly responsible for participants
- [ ] Other

Please feel free to add a cover note or any remarks to F/REC

Declarations: *

- [x] I hereby confirm having read the University of Malta Research Code of Practice and the University of Malta Research Ethics Review Procedures.
- [x] I hereby confirm that the answers to the questions above reflect the contents of the research proposal and that the information provided above is truthful.
- [x] I hereby give consent to the University Research Ethics Committee to process my personal data for the purpose of evaluating my request, audit and other matters related to this application. I understand that I have a right of access to my personal data and to obtain the rectification, erasure or restriction of processing in accordance with data protection law and in particular the General Data Protection Regulation (EU 2016/679, repealing Directive 95/46/EC) and national legislation that implements and further specifies the relevant provisions of said Regulation.

Applicant Signature: * John Carmel Aquilina

Date of Submission: * 01/08/2024

If applicable: Date collection start date

Administration

REDP Application ID ICT-2023-00034

Current Status Acknowledged

If a submitted application needs to be amended, it can be withdrawn, edited, and resubmitted, and it will retain the same reference number. There is no need to submit a new application.

L-Università ta' Malta

Research Ethics and Data Protection Form

University of Malta staff, students, or anyone else planning to carry out research under the auspices of the University, must complete this form. The UM may also consider requests for ethics and data protection review by External Applicants.

Ahead of completing this online form, please read carefully the University of Malta Research Code of Practice and the University of Malta Research Ethics Review Procedures. Any breach of the Research Code of Practice or untruthful replies in this form will be considered a serious disciplinary matter. It is advisable to download a full digital version of the form to familiarise yourself with its contents (https://www.um.edu.mt/research/ethics/resources/umdocuments/). You are also advised to refer to the FAQs (https://www.um.edu.mt/research/ethics/faqs).

Part 1: Applicant and Project Details

Applicant Details

Name: John Carmel

Surname: Aquilina

Email: john.c.aquilina.22@um.edu.mt

Applicant Status: Student

Please indicate if you form part of a Faculty, Institute, School or Centre: * Faculty of Information & Communication Technology

Department: * Department of Artificial Intelligence

Principal Supervisor's Name: * Vanessa Camilleri

Principal Supervisor's Email: * vanessa.camilleri@um.edu.mt

Co-Supervisor's Name:

Study Unit Code: * ICS5200

Course Title: * Master of Science in Artificial Intelligence [Taught and Research (Mainly by Research)]

Student Number: * 6803H

Project Details

Title of Research Project: *

Enhancing Teacher skills in handling Students Emotions using Large Language Models for the Generation of Classroom Scenarios

Project description, including research question/statement and method, in brief: *

Project Description:
This project aims to develop an AI-powered tool to assist teachers in enhancing their classroom management skills. By simulating realistic classroom scenarios featuring challenging student behaviours, the tool will provide teachers with personalised feedback and insights into effective strategies for managing these situations.

Research Statement:
This research investigates the potential of large language models (LLMs) to generate realistic classroom scenarios and analyse teacher responses, thereby offering a safe and supportive environment for teachers to practice and refine their classroom management techniques.

Methodology:
Our Class Simulation System will be used to allow teachers to generate a situation, talk to the AI agents, and get an analysis on how the teacher handled it. This system includes the following:
- Orchestrator: A fine-tuned LLM will be utilised to generate diverse classroom scenarios involving challenging student behaviours.
- Student Agent: An LLM-based agent which simulates student behaviour within the scenarios, reacting to teacher actions and exhibiting realistic emotional responses.
- Analyser: A fine-tuned LLM model will analyse teacher responses to the scenarios, providing feedback based on best practices.

Data Collection and Evaluation:
For Data Collection, we have two tools:
- Anonymous Questionnaire asking including: 1) the participants beliefs and Technology Acceptance inspired from Kriek And Stols's 2010 paper "Teachers' beliefs and their intention to use interactive simulations in their classrooms", 2) feedback on the Usability of the System with questions from the System Usability Scale (SUS), 3) feedback on the engagement of the Simulations with questions from the Game User Experience Satisfaction Scale (GUESS), and 4) a feedback box to give general feedback.
- Another Anonymised Questionnaire in which teachers / professionals in student behaviour will be given a set of generated data from the Class Simulation System, and will be asked to 1) choose the most realisitic class scenario generated from the Orchestrator, 2) assign the emotion values to a piece of text generated by the Student Agent, and 3) choose which feedback given by the analyser would be the most helpful

Will project involve collection of primary data from human participants? Yes / Unsure

Explain primary data collection from human participants:

a. Salient participant characteristics (e.g. min-max participants, age, sex, other): *

10 participants who are all teachers / teachers in training who have some teaching experience at any level of education in Malta;
5 participants who are teachers / professionals in student behaviour;
Ideally there is an equal gender balance amongst the participants;

b. How will they be recruited (e.g. sampled, selected, contacted, etc.): *

Participants will be recruited over email or social media. Their participation is entirely voluntary.

c. What they will be required to do and for how long: *

The experiments will be split into 2:

- Class Simulation System and Usability Questionnaire

The 10 participants who are teachers / teachers in training will be required to do two things:
First, the teachers will use our Class Simulation system by accessing it on our personal laptop. They will choose a scenario to handle, then go through the scenario, talking to the students, and finally they will receive feedback on their actions.
Second, they will then answer the anonymous questionnaire which includes 11 questions asking about the participants beliefs and acceptance of technology inspired from Kriek And Stols's 2010 paper "Teachers' beliefs and their intention to use interactive simulations in their classrooms", 6 questions from the System Usability Scale (SUS) template, 5 questions from the Game User Experience Satisfaction Scale (GUESS) Questionnaire, and a general feedback box in which they can write their issues with the program, and what improvements they would like to see in the future.
The entire experiment should take a maximum of 20 minutes to complete, with 15 minutes being dedicated to trying the Unity implementation and the remaining 5 minutes to fill-in the form.

- Validation Questionnaire:

The 5 participants who are teachers / professionals in student behaviour will be required to answer an anonymous questionnaire containing questions regarding the realism of the scenarios generated by the Orchestrator, the emotions detected from an AI Agent's Response, and the quality of the feedback given by the Analyser.

d. If inducements/rewards/compensation are offered: *

No rewards will be offered.

e. How participants/society may benefit: *

By combining the expertise of teachers and the power of AI, this project aims to create an innovative tool that empowers educators to create positive and supportive learning environments for all students.

f. Is the participant's identity recorded at any stage of the research (e.g. in consent forms, records, publications): *

No.

g. The manner in which you will manage and store the data and records (including consent forms): *

Data gathered from the Unity program will be used to validate the Scenarios and Feedback generated using the fine-tuned Large Language Models (LLMs) and the responses generated from the Students. The data will also give us insight on the rate-of-change in the Student Agent Emotions, and how they could be balanced better.

Data gathered from the usability form will be used to understand the usability of this system, including what the teachers think of it, any issues they had with the system, how it could be improved and what future improvements they would like to have in a system like this.

Data gathered from the validation form will be used to understand whether our 1) fine-tuned models improve the quality of the responses, and 2) Student Agents are able to correctly represent their current emotions when responding to the teacher.

Will project involve collection of primary data from animals (live non-human vertebrates/cephalopods, their foetuses and larvae, their tissue/samples and/or dead specimens of these animals)?

No

Part 2: Self Assessment and Relevant Details

Human Participants

1. Risk of harm to participants: No / N.A.

2. Physical intervention: No / N.A.

3. Vulnerable participants: No / N.A.

4. Identifiable participants: No / N.A.

5. Special Categories of Personal Data (SCPD): No / N.A.

6. Human tissue/samples: No / N.A.

7. Withheld info assent/consent: No / N.A.

8. 'opt-out' recruitment: No / N.A.

9. Deception in data generation: No / N.A.

10. Incidental findings: No / N.A.

Unpublished secondary data

11. Human: No / N.A.

12. Animal: No / N.A.

13. No written permission: No / N.A.

Animals

14. Live animals, lasting harm: No / N.A.

15. Live animals, harm: No / N.A.

16. Source of dead animals, illegal: No / N.A.

General Considerations

17. Cooperating institution: No / N.A.

18. Risk to researcher/s: No / N.A.

19. Risk to environment: No / N.A.

20. Commercial sensitivity: No / N.A.

Other Potential Risks

21. Other potential risks: No / N.A.

22. Official statement: Do you require an official statement from the F/REC that this submission has abided by the UM's REDP procedures?
No / N.A.

Part 3: Submission

Which F/REC are you submitting to? * Faculty of Information & Communication Technology

Attachments:

- Class_Simulation_-_Usability_Feedback_Questionnaire.pdf (size: 215.1 KB, uploaded on: 18/03/2025 16:35:43)
- Class_Simulation_-_Validation_Questionnaire.pdf (size: 158.3 KB, uploaded on: 18/03/2025 16:35:43)
- Class_Simulation_-_Informed_Consent_Form.pdf (size: 549.2 KB, uploaded on: 18/03/2025 16:35:43)

- [ ] Information and/or recruitment letter*
- [x] Consent forms (adult participants)*
- [ ] Consent forms for legally responsible parents/guardians, in case of minors and/or adults unable to give consent*
- [ ] Assent forms in case of minors and/or adults unable to give consent*
- [x] Data collection tools (interview questions, questionnaire etc.)
- [ ] Data Management Plan
- [ ] Data controller permission in case of use of unpublished secondary data
- [ ] Licence/permission to use research tools (e.g. constructs/tests)
- [ ] Any permits required for import or export of materials or data
- [ ] Letter granting institutional approval for access to participants
- [ ] Institutional approval for access to data
- [ ] Letter granting institutional approval from person directly responsible for participants
- [ ] Other

Please feel free to add a cover note or any remarks to F/REC

Declarations: *

- [x] I hereby confirm having read the University of Malta Research Code of Practice and the University of Malta Research Ethics Review Procedures.
- [x] I hereby confirm that the answers to the questions above reflect the contents of the research proposal and that the information provided above is truthful.
- [x] I hereby give consent to the University Research Ethics Committee to process my personal data for the purpose of evaluating my request, audit and other matters related to this application. I understand that I have a right of access to my personal data and to obtain the rectification, erasure or restriction of processing in accordance with data protection law and in particular the General Data Protection Regulation (EU 2016/679, repealing Directive 95/46/EC) and national legislation that implements and further specifies the relevant provisions of said Regulation.

Applicant Signature: * John Carmel Aquilina

Date of Submission: * 18/03/2025

If applicable: Date collection start date

Administration

REDP Application ID ICT-2025-00001

Current Status Acknowledged

If a submitted application needs to be amended, it can be withdrawn, edited, and resubmitted, and it will retain the same reference number. There is no need to submit a new application.

A.2 | Consent Form for Usability Experiment

Informed Consent Form for Participants

Name of the Researcher: John Carmel Aquilina

Title of Research: Enhancing Teacher skills in handling Students Emotions using LLMs for the Generation of Classroom Scenarios

This Informed Consent Form has two parts:

- Information sheet (to share information about the study with you).
- Declaration of Consent (for signatures should you choose to participate)

Part I: Information Sheet

Introduction

My name is John Carmel Aquilina, and I am currently pursuing a Master of Science in Artificial Intelligence at the University of Malta. This study forms part of my degree requirements and aims to contribute valuable insights into the educational sector. Specifically, I am implementing a tool designed to help educators develop effective strategies for managing students' emotions and challenging behaviours in the classroom. Through interactive classroom scenario simulations, teachers will engage with virtual students and receive feedback based on real-life teaching strategies.

Purpose of Research

The purpose of this research is to attempt to combine the expertise of teachers and the power of AI to create an innovative tool that empowers educators to create positive and supportive learning environments for all students.

Type of Research Intervention / Procedure

You will participate in a 20-minute individual trial consisting of two phases:

1. Class Simulation (15 minutes): You will interact with our Class Simulation System on a provided laptop. You may select a classroom scenario, interact with virtual students, and apply your own strategies to manage student emotions and behaviours. After completing the simulation, you will receive an analysis of your interactions, with an opportunity to ask questions about the feedback.
2. Anonymous Questionnaire (5 minutes): After the simulation, you will complete an anonymous questionnaire about your experience and provide any feedback for improvement.

Participant Selection

You have been specifically chosen to participate in this research due to your experience in teaching. We believe your perspective in handling challenging behaviour is crucial in getting feedback on the implementations within the Class Simulation System.

Voluntary Participation – Right to Refuse or Withdraw

Participation is voluntary, you do not need to take part in the research if you do not wish to. This will not affect the study adversely in any way. If you have chosen to participate, you may also voluntarily opt out of the research at any time without any consequences.

Duration

The experiment will take a maximum of 20 minutes. 15 minutes to use our Class Simulation System and 5 minutes to fill in the anonymous questionnaire.

Risks and Benefits

There aren't any risks to the participants directly. The only risks that are possible are only related to the experiment, with the data being unusable if the experiment somehow breaks.

This project can benefit teachers by giving them an area to get feedback on their methods of handling challenging behaviours and their emotions in a classroom scenario.

Confidentiality

No personally identifiable information will be collected. The only demographic data recorded will be the participant's gender. All data will be kept confidential and accessible only to the researcher and their supervisor. Data will be securely stored and retained only until the dissertation submission deadline, after which it will be permanently deleted.

Sharing the Results

We will publish the findings from this experiment in the final submission of the paper on the University of Malta's website.

Who to Contact

Researcher:

Name: John Carmel Aquilina

Email: john.c.aquilina.22@um.edu.mt

Phone Number: 99378493

Supervisor:

Name: Vanessa Camilleri

Email: vanessa.camilleri@um.edu.mt

Part II: Declaration of Consent

- ❍ I have been invited to participate in a research titled **Enhancing Teacher skills in handling Students Emotions using LLMs for the Generation of Classroom Scenarios**.
- ❍ I confirm that I have read and understood the above information, or it has been read to me, and that I agree to participate in this study.
- ❍ I have had the opportunity to ask questions about the research, and any questions I have asked have been answered to my satisfaction.
- ❍ I understand that I am free to contact the researcher or the researcher's supervisor to seek further clarification and information.
- ❍ I understand that my participation in this study is voluntary and that I am free to withdraw from the study at any time, without giving a reason and without consequence of any kind.
- ❍ I understand that all data are anonymous and that there will not be any connection between the personal information provided and the data.
- ❍ I understand that there are no known risks or hazards associated with participating in this study.
- ❍ I understand that my identity will remain anonymous in any form of dissemination, written or otherwise.

Name of Participant Signature of Participant Date

A.3 | Teacher & Expert Questionnaires

Data Gathering on Class Situations

My name is John Carmel Aquilina, and I'm a Master's Student in Artificial Intelligence at the University of Malta. Under the supervision of Dr. Vanessa Camilleri, my dissertation involves developing a system that generates class situations in a virtual class environment hosting several interactive artificial agents as students.

The motivation behind this project lies in providing a system teachers can use to refine their abilities in dealing with situations concerning students with challenging behaviours and, in general, controlling the emotions of the class to have every student in a good mood and willing to learn.

By answering this questionnaire, you will be relating to your personal experience in teaching, and your observations of how behaviours and actions may affect students' emotions within a classroom situation.

Also be informed that when answering this questionnaire all responses will be kept **anonymous** and your participation is **voluntary.** All information collected will be used for research purposes only.
If you have chosen to participate, you may also voluntarily opt out of the research at any time without any consequences.

Should you have any questions, you can contact me through:
Email: **john.c.aquilina.22@um.edu.mt** or Phone: **99378493.**

If you agree with the above and would like to participate, please continue below by giving your consent.

* Indicates required question

1. Do you give your consent? *

Mark only one oval.

- ◯ Yes
- ◯ No

Class Information

This section collects some general anonymous information on yourself

2. Which education level do you currently teach? *

Mark only one oval.

- ◯ Primary
- ◯ Secondary
- ◯ Other: _____

3. Which education levels have you previously taught? *

Tick all that apply.

- ☐ Primary
- ☐ Secondary
- ☐ Other: _____

Scenarios affecting the Emotions of Students

This section collets information on possible Negative / Positive Scenarios which can occur within a classroom.

4. **Scenario 1:** Describe a negative situation that can occur within the class when the students are waiting for a teacher to arrive for the lesson. (Please include details about student behaviours, emotional states, and potential triggers.) *

_____

_____

_____

_____

_____

5. How do you believe this situation impacts the overall mood and learning environment of the class? *

6. What strategies have you used or observed to prevent or address this type of scenario? Were these strategies effective? *

7. **Scenario 2:** Describe a situation where a student's behaviour disrupted the class. How did you respond, and what was the outcome? *

8. **Scenario 3:** Describe a situation where you successfully de-escalated a challenging student's behaviour. What techniques did you use? *

9. **Scenario 4:** Describe a time when you observed a colleague effectively manage a challenging classroom situation. What did they do that you found helpful? *

10. What do you consider to be the most common triggers for challenging behaviours in your classroom? (e.g., transitions, specific subjects, group work) *

Emotional Regulation and Classroom Strategies

This section collects information on regulating your own and other students emotions within a classroom, and the strategies relating to the regulating of emotions.

11. How do you typically manage your own emotions when faced with challenging student behaviours? *

12. What strategies do you use to promote positive emotional regulation in your students? *

13. What types of professional development or resources would be most helpful to you in addressing challenging behaviours and fostering positive classroom environments? *

14. Is there anything else you'd like to share about your experiences or strategies related to classroom management and student behaviour?

Class Simulation Usability Feedback

My name is John Carmel Aquilina, and I'm a Master's Student in Artificial Intelligence at the University of Malta under the supervision of Dr. Vanessa Camilleri.

In this form, you will be sharing your experiences with the system you just used. This feedback will help us assess the system's usability.

Also be informed that when answering this questionnaire all responses will be kept **anonymous** and your participation is **voluntary.** All information collected will be used for research purposes only.
If you have chosen to participate, you may also voluntarily opt out of the research at any time without any consequences. Should you have any questions, you can contact me through: Email: **john.c.aquilina.22@um.edu.mt** or Phone: **99378493.**

If you agree with the above and would like to participate, please continue below by giving your consent.

* Indicates required question

1. Do you give your consent? *

Mark only one oval.

- ◯ Yes
- ◯ No

Technology Acceptance

In this section, you will be answerings questions about your personal opinions on the Class Simulation and the use of AI systems in Educational settings. These questions have been were formatted according to the model discussed in Kriek And Stols's 2010 paper "Teachers' beliefs and their intention to use interactive simulations in their classrooms"

2. Perceived Usefulness of Class Simulation *

Mark only one oval per row.

	Strongly Disagree	Disagree	Slightly Disagree	Neutral	Slightly Agree	Agree	Strongly Agree	N/
The AI Class Simulations help teachers improve handling disruptive behaviours in class.								
The AI Class Simulations expose the teacher with issues which can occur during class.								
The AI Class Simulations provide helpful feedback for teachers to use in a real life class scenario.								
Talking to the Mentor helped provide clarification when needed in regards to the analysis.								

3. Perceived Compatability of Class Simulation *

Mark only one oval per row.

	Strongly Disagree	Disagree	Slightly Disagree	Neutral	Slightly Agree	Agree	Strongly Agree
The Scenario generated in the AI Class Simulation is similar to real-life classroom scenarios							
The AI Agents within the AI Class Simulation responded similarly to how real-life students responded.							
The Analyser in the AI Class Simulation generated feedback which is similar to feedback given by real life professionals.							

4. General Technology Proficiency *

Mark only one oval per row.

	Strongly Disagree	Disagree	Slightly Disagree	Neutral	Slightly Agree	Agree	Strongly Agree
I am confident in using AI systems similar to ChatGPT.							
I am confident using Computer Applications for the purpose of Educational Training.							
I can independently learn how to use new online educational tools.							
I am comfortable using various digital tools for lesson planning and teaching.							

Class Simulation Usability

In this section, you will be answering questions about the Class Simulation's Usability. These questions have been taken from the System Usability Scale (SUS) Questionnaire.

5. Class Simulation Usability Scale *

Mark only one oval per row.

	Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree
I thought the Class Simulation was easy to use.					
I think that I would need the support of a technical person to be able to use the Class Simulation.					
I found the various functions in the Class Simulation were well integrated.					
I found the Class Simulation very awkward to use.					
I needed to learn a lot of things before I could get going with the Class Simulation.					
I think that the Class Simulation can be					

useful to train teachers.

Class Simulation Engagement

In this section, you will be answers questions about the Class Simulation's Engagement. These questions have been taken from the Game User Experience Satisfaction Scale (GUESS) Questionnaire.

6. Class Simulation Engagement Scale *

Mark only one oval per row.

	Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree
I am captivated by the Simulation from the beginning.					
I think the characters in the simulation are well developed.					
I think the characters in the simulation respond realistically.					
I can clearly understand the Simulation's Story.					
I am very interested in seeing how the events in the simulation will progress.					

Class Simulation General Feedback

In this section, you will be given an area to write down any issues you had with the system, along with what improvements you would like to see. If you have no comments, just write N/A.

7. What issues did you find and improvements you would like to see in our developed system. *

Class Simulation Validation Questionnaire

My name is John Carmel Aquilina, and I'm a Master's Student in Artificial Intelligence at the University of Malta. Under the supervision of Dr. Vanessa Camilleri, my dissertation involves developing a system that 1) generates class situations in a virtual class environment hosting several interactive artificial agents as students, 2) allows teachers to act within the situation, talking to the artificial students, and 3) receive an analysis on their performance.

The motivation behind this project lies in providing a system teachers can use to refine their abilities in dealing with situations concerning students with challenging behaviours and, in general, controlling the emotions of the class to have every student in a good mood and willing to learn.

By answering this questionnaire, you will be answering questions related to the content generated, helping us understand how correct and accurate the content shown to the teachers is.

Also be informed that when answering this questionnaire all responses will be kept **anonymous** and your participation is **voluntary.** All information collected will be used for research purposes only.
If you have chosen to participate, you may also voluntarily opt out of the research at any time without any consequences. Should you have any questions, you can contact me through: Email: **john.c.aquilina.22@um.edu.mt** or Phone: **99378493.**

If you agree with the above and would like to participate, please continue below by giving your consent.

* Indicates required question

1. Do you give your consent? *

   Mark only one oval.

   - ◯ Yes
   - ◯ No

Orchestrator Validation

In this Section, you will be presented 2 Generated Scenarios. For each pair of Scenarios, you need to pick which one is most likely to occur in a real life classroom. You can also pick *Both* if both of the scenarios are likely to occur in a real life classroom, or *Neither* if neither of the scenarios are likely to occur in a real life classroom.

2. Prompt 1: During a science lesson, Ryan and Jake start whispering about their upcoming soccer match. Their conversation catches the attention of other students, leading to some disruption. The rest of the class shifts between focusing on the lesson and getting distracted by the chatter. *

vs

Prompt 2: During a mathematics lesson focused on algebra equations, two students, Liam and Ethan, start chatting softly about the upcoming school soccer match. Other classmates like Noah and Mason notice but continue working on their problems, while some, like Jacob, show signs of distraction. Meanwhile, Oliver is intently taking notes, and Lucas is focused but occasionally glances at the chatting pair.

Mark only one oval.

- Prompt 1
- Prompt 2
- Both
- Neither

3. Prompt 1: During a history lesson on the Renaissance, Alex and Ryan whisper to each other about an upcoming football game. In the middle of the class, their conversation catches the attention of some nearby classmates. Meanwhile, the rest of the class is either engaged in the lesson or trying to sneak glances at the duo. *

vs

Prompt 2: During a history lesson, Antonio and Max chat animatedly at the back of the classroom, drawing the attention of other students. James, seated nearby, struggles to concentrate, while Liam, sitting in front, turns around to ask them to keep it down. Nearby, Ethan and Gabriel exchange amused glances as they catch snippets of the conversation. Meanwhile, Ryan, Josh, and Lucas continue to focus on the lesson, seemingly unfazed by the commotion.

Mark only one oval.

- Prompt 1
- Prompt 2
- Both
- Neither

4. Prompt 1: During a literature class, Jason teases Michael about his recent presentation, making comments under his breath. Michael tries to focus but becomes visibly uneasy, while the rest of the class responds in various ways. A group discusses quietly while some pretend not to notice, hoping it will pass. *

   vs

   Prompt 2: During a group project session on a sunny afternoon, Ethan is noticeably upset about his project partner, Ryan, making offensive jokes. The class is divided into smaller groups, with some students concentrating on their tasks, while others in Ethan's group are aware of the growing tension. As whispers fill the room, Ethan considers approaching his friends for support.

   Mark only one oval.

   - ◯ Prompt 1
   - ◯ Prompt 2
   - ◯ Both
   - ◯ Neither

Student Agent Emotion Validation

In this section, you will be assigning Emotion values to the several responses generated by the AI Agent. For each Question, you have to assign a score from 1 to 100 to **any one or more** of the 7 Emotions; **Sadness**, **Surprise**, **Joy**, **Disgust**, **Anger**, **Fear** and **Neutral**. You can give a score to any number of emotion, from just 1 to all 7 depending on what you think is present in the response. A score of 0 means the emotion is not present at all, and a score of 100 means that the emotion is strongly expressed.

Important: Each emotion is scored independently, you can assign up to 100 to each emotion, your total across all emotions doesn't need to add up to 100

Answer each question with the following formatting replacing the XX with the value you would assign it:

sadness: XX, surprise:

XX, joy:

XX, disgust: XX, anger: XX, fear: XX, neutral: XX

5. That's right, Ryan! It's super cool how people started exploring art and science again! Just like a football team training hard to come back stronger! I can't wait to see what more we discover about it, just like my excitement for the game this weekend! *

6. Yeah, I think I need a little break. Just to clear my head. This whole thing feels like a lot right now, and I need some space to think and calm down. I hope things will feel better later. *

7. Seriously? I'm just trying to have a little fun. You don't have to make it a big deal. I get it, but it's stupid to act like I'm ruining everything. Can't you just let us chill for a sec? *

Analyser Validation

In this section, you will be choosing realistc advice that may be given to a teacher to manage disruptive behaviour in a classroom. Similarly to the Orchestrator Validation section, for each pair of Advice given, you need to choose which one is the most helpful. You can pick the *Both* option if the options are both equally helpful, or the *Neither* option if neither of the options are helpful.

8. Prompt 1: Your approach to handling the classroom disruption was generally effective. By initially addressing Liam and Noah in a neutral tone and asking questions, you engaged them directly, which helped to refocus their attention without escalating the situation. Although there was an unexpected moment where your emotions shifted to a brief, intense feeling of anger, you quickly returned to a neutral and composed tone. This helped maintain classroom stability, as evidenced by the students' emotional states that largely remained unchanged, with the exception of Noah who seemed slightly more stressed but ultimately responsive to your guidance. Lucas's stated need for focus was understood and was effectively redirected back to task, allowing the lesson to progress smoothly. What worked particularly well was your consistent calmness after the brief disruption, coupled with offering students a chance to discuss their enthusiasm for gaming after the lesson, managing to redirect their excitement toward a more appropriate time. This not only kept the students engaged but also showed respect for their interests. Your encouragement for Noah and others to ask questions if they felt lost was a positive strategy that reassured those who might have been anxious, like Noah, creating an environment where students felt supported and less stressed. Overall, maintaining neutrality effectively modulated the students' emotions, helping them transition back to their learning focus. *

VS

Prompt 2: You effectively managed the classroom disruption by maintaining a primarily neutral tone, which helped prevent escalation. Initially addressing Liam and Noah's enthusiasm about the video game allowed you to acknowledge their excitement without fully derailing the lesson. When you expressed a momentary intense response, possibly due to the disruption's persistence, it briefly contributed to Noah's emotional discomfort. However, you swiftly returned to a neutral and patient approach, encouraging questions and offering to start the lesson over. This likely helped reassure anxious students like Noah and refocused their attention on the material. Despite the temporary setback, your strategy to engage directly with the students involved and guide the class back to fractions proved effective. This is evident as Liam and others appeared willing to focus on the lesson once they had the opportunity to discuss their interests later. Your patience in offering additional explanation and maintaining openness to questions likely contributed to a constructive learning environment, although some students, like Noah, continued to experience stress. Encouraging communication and acknowledging external factors without letting them

dominate the session allowed most students to either sustain or regain their focus.

Mark only one oval.

- Prompt 1
- Prompt 2
- Both
- Neither

9. Prompt 1: In this scenario, your initial response to Jason's behavior showed a strong sense of urgency to protect Michael, but it was primarily driven by anger. This intensity impacted the classroom atmosphere, placing a focus on Jason's actions and potentially escalating his stress and nervousness. Over the course of the interaction, your emotional responses evolved from anger to a more neutral and understanding tone, which helped in gradually calming Jason, although his emotions initially escalated before finally showing improvement. Your transition to a calmer demeanor allowed Jason to feel acknowledged, which led him to express remorse and willingness to apologize, showing that the shift in your approach positively influenced his emotions and helped reach a resolution. One of the most effective aspects of your approach was encouraging reflection and learning from mistakes, which was evident when you acknowledged Jason's feelings and provided him the opportunity to step back and reflect. This shifted both Jason and Michael's emotional states towards calm and understanding, helping them navigate their emotions. Furthermore, offering Michael space to process the incident respected his feelings and facilitated a more inclusive resolution. For future situations, cultivating an initial response that blends firmness with empathy could help de-escalate conflicts swiftly while maintaining a supportive classroom environment. *

VS

Prompt 2: In handling the classroom situation between Jason and Michael, your initial reactions were dominated by anger, which was quite intense. While your intent to address Jason's behavior promptly was appropriate, the high intensity of anger in your responses may have escalated his defensive emotions. Jason's emotions shifted from being alert and partially excited to feeling upset and stressed, which suggests that he reacted negatively to the perceived reprimand. Your approach effectively communicated the seriousness of the issue to Jason, but it might have benefited from a more neutral or composed tone to reduce defensiveness and facilitate better understanding. Your switch to a more neutral and eventually positive demeanor towards the end of the interaction clearly helped Jason relax and acknowledge the learning opportunity, as seen by his shift towards more positive emotions. Furthermore, your ability to recognize Michael's discomfort and offer him a break was commendable, demonstrating empathy and understanding of his need for space. Michael's stress decreased slightly after he was given permission to step away, which indicates that your recognition of his feelings helped to some extent. This also reflects positively on your classroom management skills, creating a supportive environment for students to express their emotions and find solutions. Overall, while your initial

emotional responses were perhaps too intense, your shift towards a calming and constructive tone eventually created a space for resolution and learning.

Mark only one oval.

- ◯ Prompt 1
- ◯ Prompt 2
- ◯ Both
- ◯ Neither