AutoPBL: An LLM-powered Platform to Guide and Support Individual Learners Through Self Project-based Learning

Yihao Zhu*
Department of Computer Science and Technology
Tsinghua University
Beijing, China
zhuyh22@mails.tsinghua.edu.cn

Zhoutong Ye*
Department of Computer Science and Technology
Tsinghua University
Beijing, China
yezt24@mails.tsinghua.edu.cn

Yichen Yuan
Engineering
University of California, Berkeley
Berkeley, California, USA
yichenyuan@berkeley.edu

Wenxuan Tang
Nanjing University
School of Management and Engineering
Nanjing, Jiangsu, China
201870041@smail.nju.edu.cn

Chun Yu†
Department of Computer Science and Technology
Tsinghua University
Beijing, China
chunyu@tsinghua.edu.cn

Yuanchun Shi
Department of Computer science and Technology
Tsinghua University
Beijing, China
Qinghai University
Xining, China
shiyc@tsinghua.edu.cn

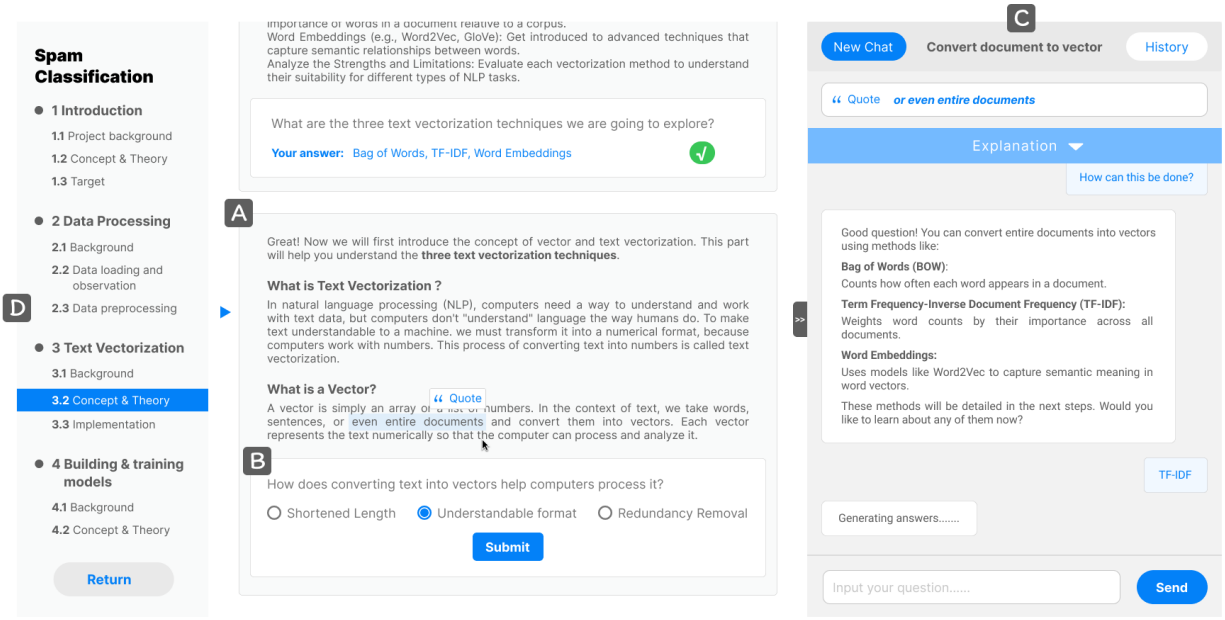


Figure 1: AutoPBL delivers an LLM guided-and-supported self project-based learning experience on an integrated GUI. In this example, a learner uses AutoPBL to learn machine learning through a *spam classification* project. (A) The tutorial content of AutoPBL is dynamically generated in *bite-sized blocks* based on a structured framework, constantly adapting to users' progress. (B) Each tutorial block is followed by a *checkpoint question*. Users must answer the question before moving on to the next tutorial block. (C) The integrated *virtual teaching assistant* enables in-context Q&A. Users can quote tutorial content and select a preset query mode to receive a helpful response without the need for extensive prompt engineering. (D) The sidebar on the left displays the tutorial framework and allows easy navigation to previous tutorial content.

Abstract

Self project-based learning (SPBL) is a popular learning style where learners follow tutorials and build projects by themselves. SPBL combines project-based learning's benefit of being engaging and effective with the flexibility of self-learning. However, insufficient guidance and support during SPBL may lead to unsatisfactory learning experiences and outcomes. While LLM chatbots (e.g., ChatGPT) could potentially serve as SPBL tutors, we have yet to see an SPBL platform with responsible and systematic LLM integration. To address this gap, we present AutoPBL, an interactive learning platform for SPBL learners. We examined human PBL tutors' roles through formative interviews to inform our design. AutoPBL features an LLM-guided learning process with checkpoint questions and in-context Q&A. In a user study where 29 beginners learned machine learning through entry-level projects, we found that AutoPBL effectively improves learning outcomes and elicits better learning behavior and metacognition by clarifying current priorities and providing timely assistance.

CCS Concepts

• **Human-centered computing** → *Interactive systems and tools*; • **Applied computing** → **Interactive learning environments**.

Keywords

AI for education, Project-based Learning, Large Language Model

ACM Reference Format:

Yihao Zhu, Zhoutong Ye, Yichen Yuan, Wenxuan Tang, Chun Yu, and Yuanchun Shi. 2025. AutoPBL: An LLM-powered Platform to Guide and Support Individual Learners Through Self Project-based Learning. In *CHI Conference on Human Factors in Computing Systems (CHI '25), April 26–May 01, 2025, Yokohama, Japan.* ACM, New York, NY, USA, 26 pages. https://doi.org/10.1145/3706598.3714261

1 Introduction

Self project-based learning (SPBL) is a combination of self-learning and project-based learning (PBL). In SPBL, learners independently follow tutorials and build projects without the guidance of a tutor (*e.g., the learner follows a 'How to develop a Snake game' tutorial to learn Python*). This approach blends the flexibility of self-learning [19, 49] and the benefits of PBL, bringing increased motivation from practical goals [35, 69] and deeper understanding from hands-on experience [35, 77]. The rise of e-learning resource platforms has fueled the popularity of SPBL, especially for computer science (CS) and AI[1][2][3][4]. However, issues with SPBL exist. In traditional PBL, human tutors monitor progress to balance learning and doing, guide learners to grasp learning opportunities (i.e., identify, grasp, and reflect on critical information), and provide timely support. This dynamic, adaptive guidance and support is difficult to replicate through even the most well-crafted tutorials [49, 70].

One emerging source of support and guidance in SPBL is ChatGPT [32], given its vast knowledge base [7, 33, 38, 68, 92], constant availability [29, 63], and ability to engage in natural language conversations [12]. While promising, ChatGPT also poses risks of misuse. In the context of PBL, ChatGPT can be misused to create entire project artifacts, bypassing the essential learning process [91], undermining critical thinking and academic integrity [32, 75, 92]. Additionally, many students struggle to craft effective prompts, limiting their ability to use LLM-based chatbots properly [13, 37, 88]. To address these challenges, research advocates for the responsible and systematic integration of large language models (LLMs) into PBL to ensure they support educational objectives without compromising the learning experience [89, 91].

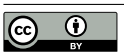
Driven by this call, in this paper, we designed a platform that guides and supports learners through self project-based learning. We focused on appropriately integrating LLMs and externalizing the educational values. Specifically, we selected computer science (CS) as the subject matter because CS lends itself well to SPBL, while most CS projects can be done on a personal computer without needing external devices or environments.

To determine the role of our platform in SPBL, we first examined the roles of human tutors in PBL through a formative study. Teaching assistants (TAs) who have closer and more direct interactions with learners are the more representative PBL tutors in CS education when compared to lecturers and tutorial authors. Thus, we conducted formative interviews with six TAs who had experience hosting PBL with fixed learning and artifact goals (contrary to exploratory PBL). The interview focused on the recurring challenges, teaching strategies, and views on ChatGPT usage in PBL. We set design goals based on their feedback. The key points were: 1) provide helpful but not end-to-end assistance, 2) monitor the progress to keep on track, 3) emphasize key learning opportunities, and 4) foster motivation for both doing and learning.

With these goals in mind, we designed and implemented AutoPBL, an LLM-powered self-PBL platform. AutoPBL is designed to help SPBL learners by 1) dynamically displaying tutorial content in a structured, block-by-block, and adaptive way, 2) offering checkpoint questions for proactive guidance, and 3) providing an in-context LLM chatbot as a virtual teaching assistant.

We conducted a counterbalanced user study where 29 beginners learned machine learning (ML) through two entry-level projects, one using AutoPBL and the other under a baseline condition. We chose entry-level ML projects because they balance learning and doing, take an appropriate amount of time to finish, and make it convenient to generate content with LLMs. Results confirm AutoPBL's effectiveness in (1) improving learning outcomes and (2) eliciting better learning behaviors and metacognition in SPBL. Our mixed-method analysis showed that AutoPBL helps learners primarily by clarifying current priorities and providing timely and appropriate assistance. Users also expressed a general preference for AutoPBL. We then analyzed users' perceptions of design features to identify areas for improvement. We wrapped up our study by discussing the needs and chances of improving LLM-generated educative content

*Both authors contributed equally to this research.
†Corresponding author.
[1]https://www.coursera.org/
[2]https://www.geeksforgeeks.org/
[3]https://github.com/practical-tutorials/project-based-learning.
[4]https://www.khanacademy.org/



CHI '25, Yokohama, Japan

ACM ISBN 979-8-4007-1394-1/25/04
https://doi.org/10.1145/3706598.3714261

and opportunities to expand AutoPBL's applicability to a broader context.

In summary, this paper makes the following contributions:

- We designed and implemented AutoPBL, a platform providing guidance and support to self project-based learning practices on machine learning.
- We systematically integrated LLM into AutoPBL to align with the goals and values of PBL derived from a formative study.
- We used a mixed-design user study to evaluate AutoPBL's effectiveness in improving learning and metacognitive outcomes and provide insights into the underlying mechanisms.

2 Related Works

2.1 Self-Learning, Project-based Learning and SPBL

Our research builds on the concept of self project-based learning (self-PBL or SPBL). SPBL combines elements of self-directed learning and project-based learning [49, 77], both of which have well-established traditions in educational research.

Self-learning (or self-study, self-education) is an approach where students learn at their own pace without direct guidance from a human instructor [61, 94, 95]. This flexibility has made self-learning popular among individuals who lack access to traditional courses or require a flexible learning schedule [8, 19, 49]. Its popularity is extreme in fields rich with online resources, such as programming, software usage, and machine learning [24, 39, 48].

Project-based learning (PBL) is a student-centered pedagogy where students learn and apply ideas in a specific project context, eventually producing an artifact [34, 35]. Rooted in constructivism, PBL emphasizes learners' active construction and reconstruction of their understanding through experiences and reflection [6, 35, 77], rather than passive information intake. PBL fosters learning motivation by amplifying the relevance of what they are learning and doing [6] and letting learners easily see the productions of their efforts [35, 69]. PBL has also been shown to promote creativity, problem-solving, and collaboration [22].

Integrating self-learning with project-based learning (SPBL) is effective because self-learning makes PBL flexible, while PBL boosts motivation for self-learners [43, 49]. Although SPBL often lacks key elements like team collaboration and self-direction found in traditional PBL [36, 43], it still promotes positive learning and metacognitive outcomes [49, 72, 77].

However, without a tutor, SPBL learners often struggle with self-regulation and the lack of assistance, amplifying the obstacles in executing PBL. SPBL learners may lose motivation [4], focus [23, 65, 70] and desire for exploration[20, 21, 62] in the process. Learners need timely reminders on learning opportunities [21], confirmation of cognitive progress [2, 34] and individualized support [34, 60]. Existing platforms focus on enhancing the quality of instructional materials [66], developing interactive learning platforms[5][6][7], and fostering learning forums[8][9]. However, these platforms demand intensive human effort and are restricted to tutorial learning [43, 49]. Therefore, in this paper, we set out to build a platform that intelligently provides much-needed personalized guidance and support to SPBL learners.

2.2 LLMs in Education

LLMs are increasingly being used by teachers and students alike for content generation and any-time chats [32, 75]. The rising adoption comes from their knowledge, naturalness of interaction, and scalability. First, LLMs have a solid grasp of entry-level domain-specific knowledge across diverse fields, including language [29], medicine [38], data science [68, 92], and, crucially for our study, coding and machine learning [7, 33, 42]. Retrieval augmented generation (RAG) [40], where the LLM is supplemented with an external knowledge base, has been shown to improve the accuracy of domain responses further [51]. Second, they generate responses perceived as natural and appropriate in tone [33], sometimes even more readable than those provided by human instructors [12]. Finally, LLM-based applications are easily scalable. Through prompt engineering, LLMs can generate diverse forms of educational material that are adaptive to learners' personalized needs [29, 63].

However, the responsible and accessible application of LLMs in educational settings presents a significant challenge. Misuse like end-to-end solution generation may inhibit learner's critical thinking [32, 75, 92] and compromise academic integrity [33]. Moreover, LLMs can generate false or misleading information in a convincing manner due to hallucination [25, 30, 84] (e.g., generating plausible but nonexistent academic references [50]). Hallucinations are particularly harmful to entry-level learners, who may struggle to identify false information in AI-generated content [53]. Finally, prompt engineering poses a challenge to non-expert users to efficiently use LLMs because they may be unfamiliar with domain terminologies [57, 93] or lack query skills [13, 88]. To address these drawbacks, we paid attention to ensuring robust and educative content generation and enhancing usability during interaction.

2.3 Intelligent Tutoring Systems

Intelligent Tutoring Systems (ITS) are automatic systems that track learner progress and provide adaptive support and guidance [56] like human instructors [49, 64].

One widely studied form of ITS recommends learning materials, such as exercises and instructions, based on knowledge tracing (KT) [14, 26, 41, 76], cognitive diagnosis (CD) [15, 79, 83] and other data mining [78] or rule-based algorithms [90]. These algorithms model learners' proficiency using past interactions with the system [1]. A systematic implementation using this approach is OATutor [58], an ITS that uses Bayesian KT [11] to recommend exercises based on the student's mastery of relevant skills. However, these methods follow a linear representation of learning context [86] and depend on pre-annotated resources for recommendation [1]. These weaknesses hinder comprehensive observations of learners and the efficacy of adaptive tutoring in complex and unstructured learning processes like PBL.

[5] https://www.geeksforgeeks.org/
[6] https://codecrafters.io/
[7] https://www.freecodecamp.org/
[8] https://www.coursera.org/
[9] https://www.khanacademy.org/

Large language models (LLMs) have opened new possibilities for ITS through analyzing unstructured context and generating adaptive and diverse instructional content [59, 71]. LLM-powered ITSs take in the learning materials used [87] and learners responses in conversations [10, 59] and quizzes [86] for analysis. The system then generates adaptive instructions and Q&A answers [42], or externalizes pedagogy like Socratic teaching [10, 17] and learning-by-teaching [31, 67]. In particular, LLM shows promise in computer science-related ITSs, especially in improving clarity and engagement [42], demonstrating the rationality of our user study scenario. While promising, the effectiveness of some conversational LLM-based ITSs is tested on datasets and simulated students [10, 44, 47]. Empirical studies on actual students drew inconsistent conclusions on LLM-based ITSs' effect on learning outcomes [31, 59, 87]. Thus, we seek to comprehensively evaluate AutoPBL's effectiveness in assisting learners using mixed methods.

3 Formative Study

To determine the form of guidance and support we should include in our SPBL platform, we examined the roles of human PBL tutors in sessions with fixed learning and artifact goals (contrary to exploratory PBL). The interview focused on the goals, recurring challenges, and teaching strategies of PBL. We also asked interviewees for their views on students using ChatGPT in projects. From these discussions, we derived design goals for the platform.

3.1 Formative Interview

We selected TAs instead of experienced professors (who primarily serve as lecturers) as formative interviewees because PBL tutors must closely interact with PBL learners. We recruited six TAs from a university CS department's TA group chat, all with experience in holding PBL activities with fixed learning and artifact goals (see Table 1). The TAs (age = 23.75 ± 0.76, five males and one female) served 2.5 semesters as TAs for their respective courses on average. Three of them were the original designers of the project, while the others made incremental improvements based on existing projects. They were generally responsible for formative assessment, Q&A, and the final evaluation and grading. However, only two TAs were tasked with hosting proactive discussions with students. Each TA was interviewed individually by a researcher for about an hour via video conference.

At the start of each interview, we asked TAs to describe the projects' content, process, requirements, and teaching goals. Their teaching goals align with common PBL objectives, such as learning through doing, connecting knowledge with experience, and making sense of the process. They were then asked to share their experiences and insights in three specific areas:

- Q1: *What were the main challenges and issues you encountered while hosting project-based learning activities?*
- Q2: *What specific arrangements or project design did you implement to achieve the teaching goals?*
- Q3: *To what extent were students allowed to use generative AI, like ChatGPT, in the projects you hosted?*

We followed up with additional questions based on the TAs' interview responses. We also asked them to provide example materials of the arrangements and designs mentioned in Q2. The interviews were audio recorded, transcribed, and open-coded [9] by two researchers. The research team then identified recurring themes related to the TAs' methods and ideas for adequate support and guidance in PBL, leading to four design goals for AutoPBL.

3.2 Design Goals

We synthesized four design goals that AutoPBL should meet.

G1. Provide helpful but not end-to-end assistance. While improving tutorial materials by providing scaffolding (T4, T6) and hints (T2, T3) can be a cost-effective way to assist students given high faculty-student ratios (T2, T4, T5), TAs noted that static tutorials cannot address the need for adaptive learning and personalized assistance, especially in diverse classrooms (T2, T3). Moreover, referring to Q&A and discussion, TAs noted that students often avoid seeking help because they do not know what to ask (T2) or are concerned about privacy issues (T3). ChatGPT is widely regarded as a suitable Q&A substitute for providing knowledge and explanations (T3, T5, T6). However, all TAs cautioned against using ChatGPT as an end-to-end tool to finish key parts of the project. They suspect that direct solutions would not help students develop deeper understanding (T2, T3) or construct their own knowledge (T6)—both essential goals in PBL. For instance, T6, who focuses on developing system engineering skills, warned, '*You'll struggle later if you rely on ChatGPT's code without fully understanding how the system works.*' Similarly, T2, who emphasizes user-centered design thinking, stated, '*Using ChatGPT to write code is acceptable, but for tasks requiring independent thought, reliance on GPT undermines the course's goals.*' Some TAs expressed more flexibility with ChatGPT's use for non-core tasks (T1, T4) and considered effective GPT use as part of students' skills (T4, T6).

G2. Monitor the progress to keep on track. Failures in PBL often stem from going off track (T2, T4). Due to time limits, TAs cannot closely monitor each group's progress and provide corresponding guidance (T1). They break down the project and set specific checkpoints (T1, T2, T6) or subtasks (T4, T5) to address this. For example, T1 provided a result visualization script for self-checking implementation correctness, while T4 suffixed each tutorial session with a conceptual self-check question. These milestones allow them to assess whether students' projects and learning still align with the objectives. Meanwhile, TAs design project progression tailored to the project's characteristics. For example, T3 and T5 use a "build around the core" approach, where students implement the core functionality and add features incrementally. Others follow a modular (T4, T6) or step-by-step design (T1, T2). As we intend to support the entire SPBL process with LLM, we can more frequently and proactively monitor progress to keep learners on track.

G3. Emphasize key learning opportunities. Achieving learning goals requires identifying key learning opportunities. TAs guide students to identify important information (T4), take critical reflections (T2, T3, T4), and write core code in person (T1, T4, T5). Without this, students may complete the project without fully understanding the subject (T6). For example, T4 mentioned that if he did not require students to read the documentation of the C++ *rand()* function, they would miss the concept of pseudorandomness. The problem of missing learning opportunities is addressed through

Course & Project Information			Duties in Supporting PBL					
TA ID	Course	Project	Project Design		Procedural Tasks			Evaluation
			Design	Improve	Discussion	Formative Assessment	Q&A	
T1	Pattern Recognition	*Image Classification with Large Margin Nearest Neighbor*	✓			✓	✓	✓
T2	Human-Computer Interaction	*Validation of Fitts' Law*		✓	✓		✓	✓
T3	Data Structures and Algorithms	*Write a Hash Table*	✓			✓	✓	✓
T4	Introduction to Programming	*Write a "Snake" Game*	✓			✓	✓	✓
T5	Distributed Systems	*Build a Toy Distributed System*		✓		✓	✓	✓
T6	Principles of Computer Composition	*Build a RISC-V CPU on FPGA*		✓	✓	✓	✓	✓

Table 1: The formative interview involved 6 CS course TAs with experience hosting PBL activities with fixed learning and artifact goals. We list the courses they supported, the projects they hosted, and the duties they took.

methods such as posing reflective questions (T4), asking the students to report observations (T3, T6), and providing supplementary reading materials (T1).

G4. Foster motivation for both doing and learning. PBL designers aim to motivate students by introducing new learning topics (T1, T3, T4) or simulating the process of building real, usable products (T5, T6). However, students often shift their focus to the grading criteria, undermining their original motivation for both doing and learning (T1, T6). Such unbalanced learning can result in disengagement from meaningful work or learning. Tools like ChatGPT, which can generate correct code or answers directly, may exacerbate this trend (T1, T4). To maintain a balance between "doing" and "learning," TAs must continuously adjust the grading scheme to steer students toward the intended learning objectives (T6). In SPBL, we anticipated less pressure from grading but recognized the ongoing need to remind learners to maintain balanced motivations.

Resolve teamwork deadlocks. The TAs also claimed that they pay much attention to instructing students for better teamwork (T2, T4, T6). In group projects, students find dividing the project into sub-tasks and then assigning them to team members challenging (T4). TAs often have to join the group discussions to resolve the deadlock (T2, T6). We **did not** consider this point in our design because AutoPBL focuses on individual learners engaged in self project-based learning. However, we discussed extending AutoPBL to collaborative settings in Section 7.2.

4 AutoPBL: Design and Implementation

As illustrated in Figure 2, we designed various features to achieve the design goals (**G1**-**G4**) and improve usability (**U**). AutoPBL features three systematically integrated elements. Tutorial serves as the backbone (**G3**) while checkpoint questions and virtual TA provide proactive (**G2**, **G3**, **G4**) and responsive (**G1**) assistance, respectively.

Tutorial follows a pre-generated project framework to guarantee coherency (**G2**). The tutorial is presented in bite-sized blocks, while the framework is shown in the sidebar (**U**). Checkpoint questions are derived from a taxonomy tailored to SPBL that is designed to proactively engage learners with tasks on varying aspects of SPBL, including learning, doing, and high-level thinking (**G2**, **G3**, **G4**). The virtual TA enables in-context and educative Q&A (**G1**), with features such as quote-and-ask and preset inquiry modes supporting intuitive question formulation (**U**).

These three design elements are systematically integrated through an underlying LLM-based multi-agent auto-generation system. The tutorial provides context for checkpoint question generation and the virtual TA. Meanwhile, the learner's responses to checkpoint questions are used to assess progress and facilitate adaptive tutorial generation and virtual TA responses.

4.1 Structured, Block-by-block Adaptive Tutorial

AutoPBL tutorials follow a pre-generated **two-level project framework**. This framework is shown to learners as a table of contents

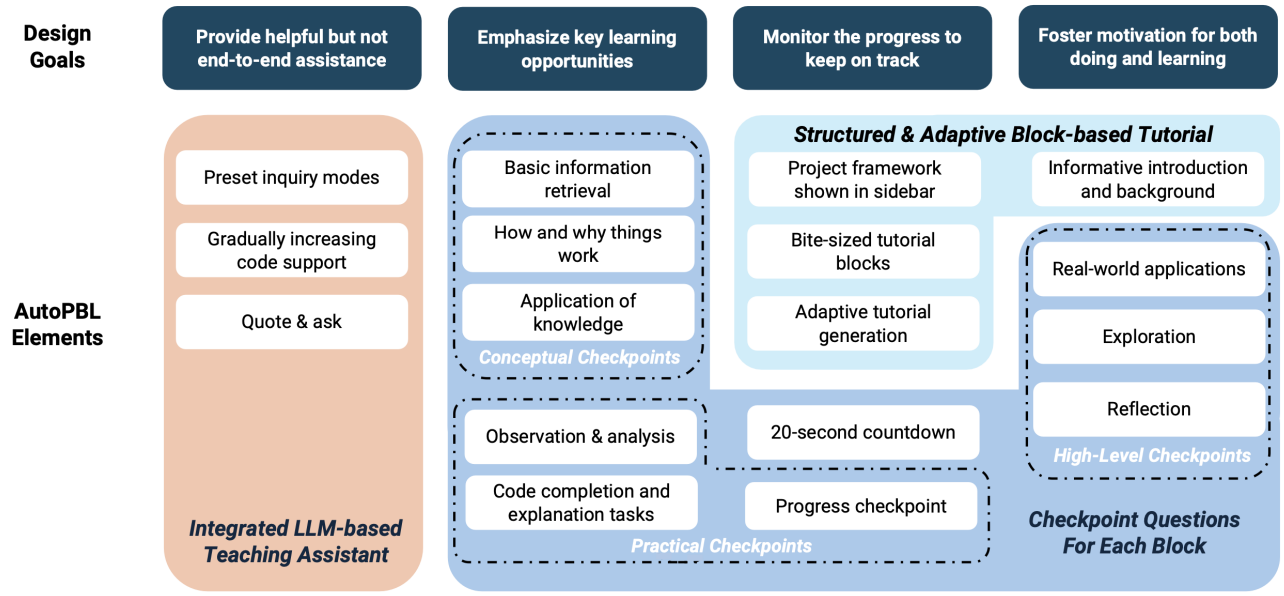


Figure 2: This figure outlines the design elements used in AutoPBL to address the design goals formed in the formative study. Each column shows a design goal and the corresponding design elements to address these challenges. The elements are also grouped by their overarching system feature (e.g., tutorial, checkpoint questions, virtual TA). Specifics about checkpoint questions can be found in Figure 3.

on the sidebar and can be used to navigate, similar to popular tutorial platforms[10][11]. The main content of the tutorial is presented as **bite-sized blocks**, each containing a single learning or doing task. Once the learner finishes a block, the next one will be adaptively generated according to the learner's progress and response to the checkpoint question (see Section 4.2). For example, a correct answer would trigger encouragement and enable progress to the next task, while an incorrect answer triggers a review of previous content.

Two-level Project Framework. Each project is divided into steps (e.g., data pre-processing), which are further divided into fine-grained sub-steps (e.g., data cleaning and data augmentation). This framework offers a clear overview of the project. Moreover, to foster interest and motivation, we included sub-steps that provide background information about each step.

Bite-sized Blocks. AutoPBL presents each sub-step of the tutorial as bite-sized blocks, each containing a single learning or doing task. This is a form of microlearning [54, 74], and its effectiveness is backed up by cognitive load theory [73] and the technique of chunking [52]. This design seeks to make the material more accessible and easier to digest (**U**) and help maintain the learner's attention throughout the project (**G4**).

4.2 Providing Guidance through Checkpoint Questions

In AutoPBL, each block is followed by an adaptively generated checkpoint question, similar to the project checkpoints mentioned by TAs in the formative study. The learner must respond to the checkpoint question before progressing to the next block. This design is informed by proactive tutoring strategies like using formative assessment to assess learning progress [5], and leveraging pre-questioning to guide students' attention [55]. The checkpoint questions are derived from a **taxonomy** tailored to different learning processes within SPBL. Each question is given in one of two forms: multiple-choice or short-answer. Finally, we included a 20-second timer before revealing the checkpoint question to prevent learners from skimming only for question-related information.

Taxonomy of Checkpoint Questions. We designed three categories of checkpoint questions - **conceptual**, **practical** and **high-level** questions about the project as a whole. Each category is further broken down into three question types, as shown in Figure 3. Conceptual questions require the learners to reflect upon what they have learned, how things work, and how to apply the knowledge in practical scenarios (**G3**). Practical questions raise awareness on implementation by checking the learner's progress, asking the learner to observe and analyze, and assigning coding tasks (**G2**, **G3**). Finally, high-level questions encourage reflection and metacognitive thinking while fostering motivation (**G4**).

[10] https://www.runoob.com
[11] https://www.khanacademy.org

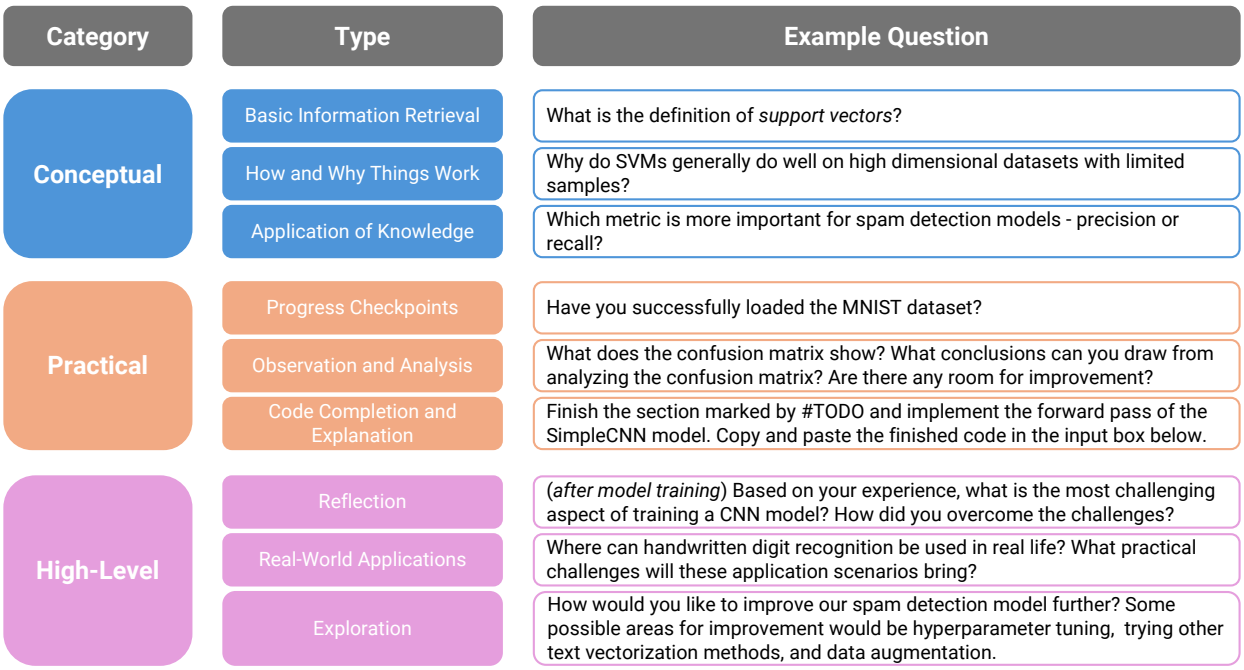


Figure 3: Taxonomy of checkpoint questions in AutoPBL. We included three categories of checkpoint questions relevant to the "learn by doing" process in PBL. Specifically, conceptual questions focus on the "learning" part, while practical questions concern the "doing" part. In addition, high-level questions are designed to trigger reflection and exploration beyond the project itself.

4.3 Chatbot as a Teaching Assistant

In AutoPBL, learners can turn to an integrated LLM-powered virtual TA for needed assistance. The virtual TA is **context-aware** (i.e., aware of the learner's progress) and prompted to avoid providing end-to-end solutions, providing **progressively increasing assistance** instead (**G1**). Moreover, users can **quote** specific content and select **preset inquiry modes** when asking the virtual TA, reducing the need for prompting (**U**).

Context-Aware Support. The virtual TA considers the tutorial content, the learner's responses to checkpoint questions, and any quoted material when generating a response. For example, when the learner quotes *"data augmentation"* in the *digit recognition* tutorial and asks for an explanation, the virtual TA will reference project-specific examples, such as *"rotating and zooming the digits to introduce diversity to the training set."* In addition, the assistant encourages learners during challenging tasks such as learning math and coding to mitigate frustration.

Progressively Increasing Assistance. The virtual TA is designed to provide progressively increasing assistance. Initially, it provides hints and references, offering complete solutions or runnable code only after multiple failed attempts by the learner to solve the problem. This encourages the learner to think and code by themselves instead of copying end-to-end solutions given by LLMs (**G1**).

Preset Inquiry Modes. We also included four built-in inquiry modes for the virtual TA - *explain*, *debug*, *generate a quiz*, and *visualize*. Users can select one of the modes to get a more directed response without detailing requirements (**U**). For example, the user would receive a 2-D visualization of a line separating clusters of dots when asking *"what is a decision boundary"* in *visualize* mode.

4.4 Implementation

AutoPBL's fully automated generation workflow is implemented as an LLM-based multi-agent system using OpenAI's GPT-4o API, as shown in Figure 4. The multi-agent design makes it easier to define the roles GPT-4o plays in AutoPBL and synchronize context through inter-agent information exchange. The prompts are provided in Appendix F.

Tutorial Generation Workflow. When creating a project, the **Project Designer** Agent generates the project framework according to the project requirements and learner profile. The **Block Content Generator** produces content based on the project framework, a summary of previous steps generated by **Summarizer** (except for the first block when there are no preceding blocks, same below), the learner's answer to the prior block's checkpoint question, and the learner's profile. Once the tutorial part of a block is generated, the **Checkpoint Question Agent** crafts a checkpoint question aligned with the new block's content. We prompted this

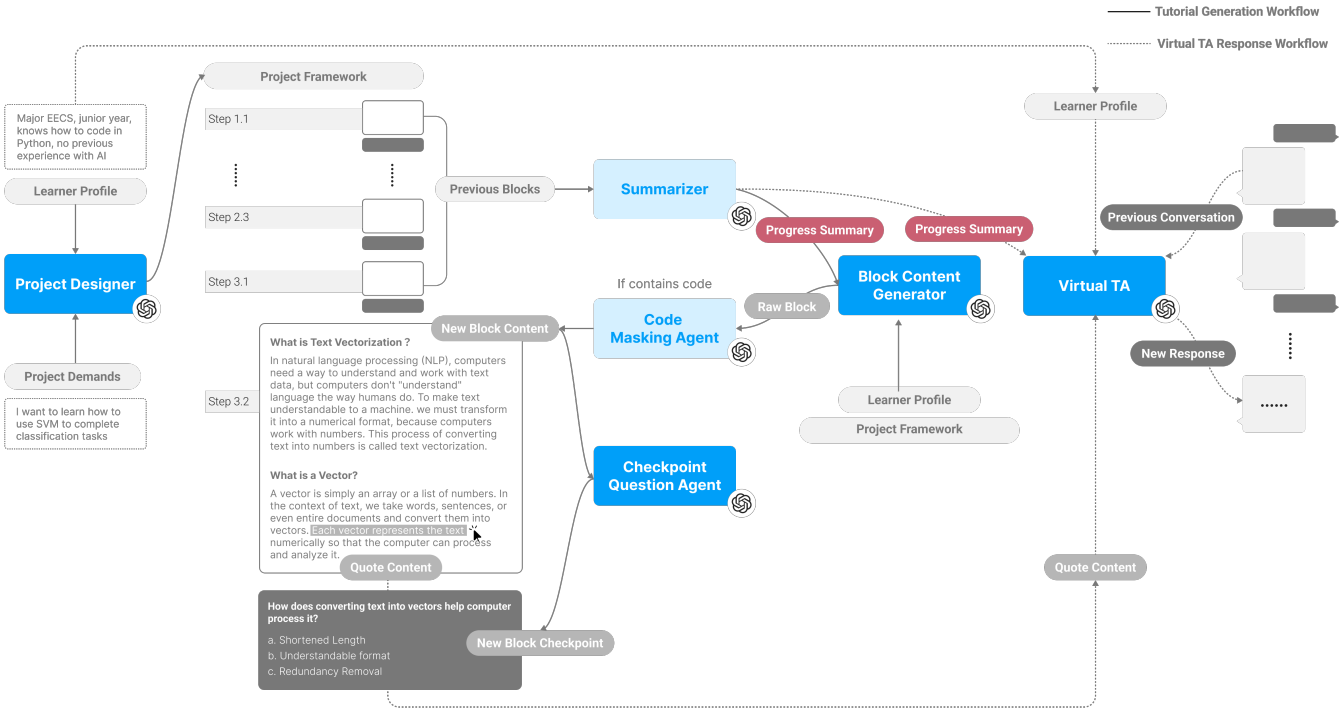


Figure 4: The multi-agent system implementation of AutoPBL. This graph illustrates how AutoPBL generates seamless SPBL content in block-based tutorials and answering inquiries. The *Project Designer*, *Block Content Generator*, *Checkpoint Question Generator* and *Virtual TA* are directly involved in supporting the features mentioned in Sections 4.1, 4.2, and 4.3. The *Summarizer* is crucial to the efficiency and speed of GPT-4o API calls by reducing context length. The *Code Masking Agent* post-processes the contents generated by *Block Content Generator* to remove excess code that gives away the solution of code completion tasks, mitigating a persistent problem of GPT-4o that possibly stems from alignment attempts to improve code generation capabilities.

agent to choose from the nine-question types in Figure 3 according to the current block's learning or doing task and generate a draft question. The Checkpoint Question Agent then refines the question through chain-of-thought prompting [81], improving relevancy and quality. Specifically, in blocks with code completion tasks, we observed that GPT-4o persistently generates full, runnable code, ignoring instructions in prompts. As a solution, we introduced a **Code Masking Agent** to post-process each block's content, removing any excess code to prevent AutoPBL from inadvertently giving away code completion answers. This phenomenon shows that productivity-oriented LLM alignment often goes against the desired behavior in educational contexts (see detailed discussion in Section 7.1).

Virtual TA Response Workflow. When processing learner queries, the Virtual TA receives a summary of the learner's progress from Summarizer, the contents of the block currently being read by the learner, and the quote content. The Virtual TA then generates a context-aware response based on the educational principles detailed in Section 4.3. When the learner selects preset question modes, we put the learner's inquiry into the corresponding prompt template, reducing the need for the user to craft prompts themselves.

Adaptability of Implementation. In AutoPBL, all contents are generated by GPT-4o since it is competent under the ML topics we chose (see Section 5.1.2 for technical evaluation). However, as the multi-agent system and prompts are not topic-specific, we assume that simply modifying the topic indicator word (e.g., changing "Machine Learning" to "C++ Programming") in prompts can adapt AutoPBL to more context, as long as the LLM is competent in that area or supplemented with external knowledge through techniques such as RAG [40].

5 User Study

We conducted a user study to evaluate AutoPBL's effectiveness in supporting self project-based learning (SPBL). This study focused on the following three research questions:

RQ1. Can AutoPBL improve learning outcomes in SPBL?

RQ2. Does AutoPBL promote beneficial learning behaviors and metacognition?

RQ3. How do learners perceive (different features in) AutoPBL?

We included a baseline condition to compare the effectiveness of AutoPBL with current SPBL practices. Participants were tasked with learning machine learning (ML) through two projects, one under each condition. An online coding workspace was provided for all the code-related tasks. We evaluated the learning process and outcome through quizzes, questionnaires, and semi-structured interviews during the experiment sessions.

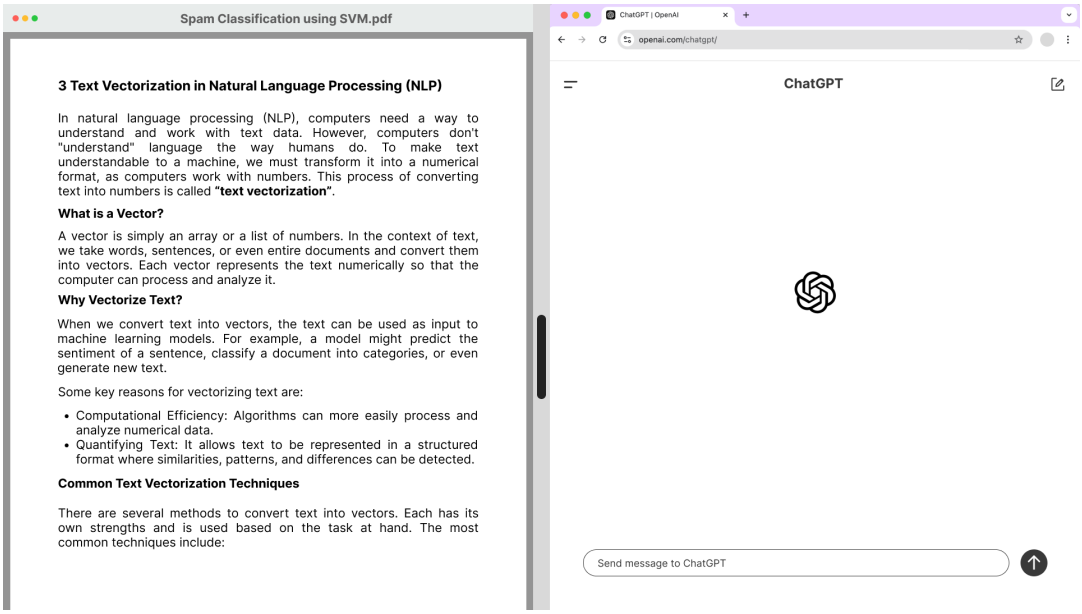


Figure 5: Illustration of Baseline condition. In the Baseline condition, participants followed static tutorials and were supported with an LLM chatbot attached to OpenAI's GPT-4o API (same as the API used in AutoPBL).

The study involved 29 participants new to ML. To facilitate fair comparison, we adopted a counterbalanced user study design (see Fig. 6). The entire procedure lasted about 3.5 hours, with participants receiving compensation equivalent to 50 US dollars. The following sections detail the experiment setup, participant recruitment and filtering, study design, and evaluation.

5.1 Experiment Setup

Before our user study, we validated that the AutoPBL generation flow could support the two selected projects. We then designed a Baseline condition to compare with AutoPBL and prepared a programming workspace for all participants.

5.1.1 Machine Learning Projects. We selected two projects for the user study based on the following criteria: (1) they should be widely-used entry-level machine learning projects, ensuring GPT-4o is capable of generating tutorials and answering questions accurately, and (2) the projects should differ significantly in task type and concepts involved, preventing repetitive learning experiences. After reviewing several options, we chose *Digit Recognition* and *Spam Classification*.

Digit Recognition involves implementing a convolutional neural network (CNN) to classify handwritten digits in the MNIST dataset [16], while *Spam Classification* requires using a support vector machine (SVM) to separate spam from ham messages in the SMS Spam dataset [3]. These projects are commonly included in beginner machine learning courses like Stanford CS229[12]. They represent two different domains in AI - computer vision and natural language processing, and use different models: CNN and SVM, respectively.

[12]https://cs229.stanford.edu/

5.1.2 Technical Evaluation. To examine GPT-4o's competence on the two projects, we used AutoPBL prompts to generate tutorial and checkpoint questions, with five novice learner volunteers learning with the materials, resulting in 291 content blocks. As all participants' native language is Chinese, the main content of the tutorial was prompted to be in Chinese (same in the formal study). Two researchers reviewed the content and found that 94.8% of the blocks did not contain errors. There are two types of errors. One is generating irrelevant questions that misreference previous blocks. For example, GPT-4o occasionally asks learners about the history of CNN when learners have progressed to the algorithm details. Another is overlooking learners' over-generalization in its assessment. For example, towards the question *"How does SVM work?"* GPT-4o may accept *"By classifying data accurately"* as correct. We assumed that these two types of errors stem from LLM's weakness in long-context generation [45] and being critical [85]. We asked the users to press the *regenerate block* button as a quick repair measure when these errors occur in the user study.

5.1.3 Baseline Condition. To compare AutoPBL with typical SPBL workflow, we designed a Baseline condition. In this condition, learners followed static project tutorials and used a standalone ChatGPT for assistance (Fig. 5). We first generated a fixed tutorial framework for each project to ensure content consistency between *AutoPBL* and *Baseline*. Then, using the same content generation prompts as in AutoPBL (see Appendix F.2), we instructed GPT-4o to generate the tutorial content, which was then compiled into a webpage. For a unified experience with LLM support, we provided access to GPT-4o through an LLM chatbot frontend that resembles the ChatGPT interface.

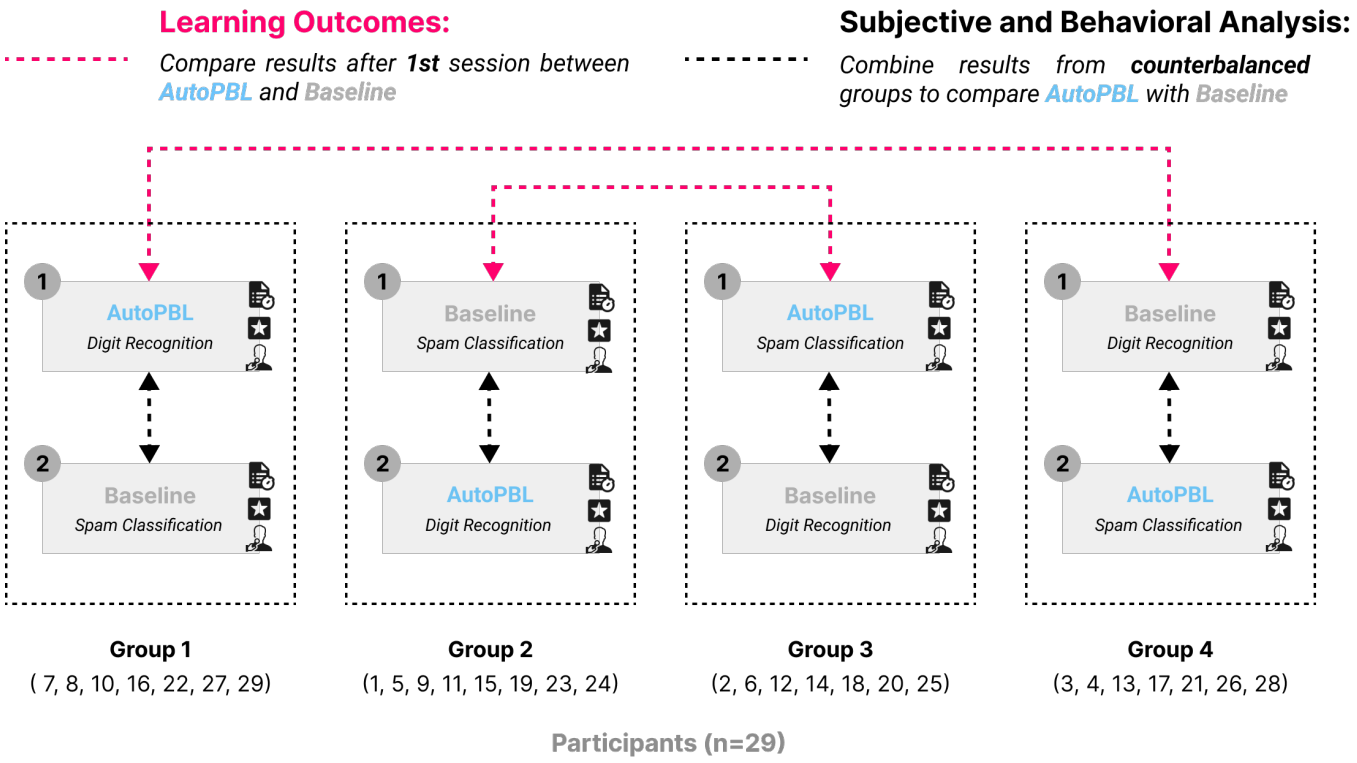


Figure 6: We adopted a counterbalanced design with four groups, where participants completed SPBL on two projects under both conditions in different orders. Twenty-nine participants were randomly and equally assigned to these groups and completed two study sessions based on their group assignment. A quiz, questionnaire, and interview followed each session. For the learning outcome comparison, we used only the results from the first session to remove the influence of learning gains. We combined data from all groups and sessions for subjective and behavioral analysis. We believe the counterbalanced design mitigates the impact of irrelevant variables.

5.1.4 Programming Workspace. In both AutoPBL and the Baseline condition, participants completed their programming tasks using Jupyter Notebook, an interactive programming tool ideal for beginners. The service was hosted on our server. All necessary packages and GPU drivers were pre-installed to streamline the experiment process.

5.2 Participants

We recruited 32 participants from campuses through advertisements on social media. Three dropped out at the entry stage for scheduling conflicts, leaving 29 valid participants (17 male, 12 female) with an average age of 20.5 years (std = 3.9).

Participants were required to have basic Python programming and debugging skills and prior knowledge of linear algebra and calculus. To ensure this, we validated their self-reported qualifications with three questions (See Appendix A.2), all of which had to be answered correctly.

We excluded applicants with prior knowledge of the project topics. They were asked to report any machine learning experience and solve four multiple-choice questions on related concepts (See Appendix A.3). We filtered out those who 1) had experience with machine learning projects, 2) explored relevant topics, or 3) answered more than one multiple-choice question correctly.

Participants also completed a 5-point Likert scale survey on their confidence in self-directed project-based learning (SPBL) and their proficiency in using LLM chatbots like ChatGPT for learning support (see Appendix A.1). Confidence was moderately low (average = 2.3, std = 0.6), and chatbot usage proficiency was neutral (average = 3.0, std = 0.89). There was no significant difference between the four experimental groups (see the following section) in SPBL confidence (one-way ANOVA, same for next, $p = 0.92$) or chatbot usage tendency ($p = 0.88$).

5.3 Study Design

The main challenge in our study design is the inevitable changes in knowledge state and fatigue levels after each SPBL session. To address this, we used a counterbalanced design where participants worked on two projects in two SPBL conditions but in varied orders. Participants were randomly assigned to one of four groups, determined by the order of using the two conditions and learning two projects.

As illustrated by Fig. 6, this design allows for a fair comparison between AutoPBL and the Baseline condition. By using only the results from the first session, we ensure an even-handed comparison of learning outcomes, as participants started as complete novices. For subjective and behavioral analysis, we combined results from both the first and second sessions, relying on the counterbalanced design to minimize the impact of irrelevant variables such as knowledge changes and fatigue.

5.4 Procedure

Out of twenty-nine experiments, six were conducted in person and twenty-three via video conference due to participants' difficulty traveling on-site. Online participants were required to find a quiet, interruption-free environment, while on-site participants were provided with private office rooms. All experiments were conducted on participants' personal computers with internet connection. We obtained participants' consent to record audio and screen activity during each session.

At the start of the first session, we introduced the Jupyter Notebook, ensuring participants understood how to write, run, and organize code cells. Next, we introduced the learning condition. In AutoPBL sessions, participants were guided through AutoPBL's key features—such as tutorial block generation, checkpoint questions, integrated virtual TA, and the table of contents in the sidebar. In Baseline sessions, we sent participants URLs for the static tutorials and LLM chatbot and asked them to learn at their own pace. Using external resources was allowed for both sessions as long as we were informed. At the start of each session, we emphasized that the goal was to 1) complete the project and 2) learn through the process. They were also informed of a closed-book quiz after the session.

Sessions ended once participants completed the projects (confirmed by the correct visualization of validation) and felt they learned everything they could. AutoPBL sessions typically lasted about 110 minutes, including GPT-4o API response time (around 25 minutes per participant), while Baseline sessions averaged 58 minutes. After each session, participants took a quiz, filled out the questionnaire, and participated in the interview. A minimum 15-minute break was required between sessions to allow for some rest.

5.5 Evaluation

5.5.1 *Evaluation Metrics.*

Quizzes. To assess the learning outcomes of the two SPBL sessions, we invited a teaching assistant from the university's CS department to design quiz questions. He crafted ten questions for each project: four on conceptual knowledge, three on execution interpretation, and three on machine learning cognition. Of the ten questions, seven were single-choice, and three were multiple-choice. All of the questions assess learners' understanding of concepts or practical skills. The questions were reviewed and refined to eliminate ambiguity and errors. The final version of the quizzes is provided in Appendix B. We examined the static tutorials for Baseline and the AutoPBL tutorials generated in technical evaluation in 5.1.2 and ensured that everything tested by the quizzes was covered in tutorials.

Questionnaires. To gather quantitative data, we used a questionnaire consisting of three (for Baseline) or four (for AutoPBL) sections. The first section focused on **perceived learning outcomes**, including understanding concepts, practical skills, and machine learning as a whole. Five questions on **metacognition in LLM-assisted SPBL** assessed the quality of human-AI interaction and confidence in future learning ML with LLMs, drawing partially from the work by Zheng et al. [91]. Furthermore, we used NASA-TLX to measure **cognitive load** during the learning process and adapted usability questions from previous research [80] to evaluate user experience. For AutoPBL sessions, we added a section where participants rated their **preference for various features of AutoPBL** and assessed the **clarity and effectiveness of checkpoint questions**. All ratings were done on a 7-point Likert scale. The questionnaire's content is presented in Appendix C.

Semi-structured interview. After each experiment session, we conducted a semi-structured interview to gather participants' feedback on their perceived gains and learning processes. We began by asking about their perceived gains in knowledge, practice, and general understanding (RQ1). Next, we asked them to reflect on their learning process, including workflow, interactions with the LLM, and mental state (RQ2). Finally, we asked about their overall learning experience, focusing on the effects of specific AutoPBL features (RQ3) and any challenges they faced. After the second session, participants also compared the two learning experiences. We further explored additional topics, such as applicability to a broader range of learning activities and potential improvements, when raised by participants. The interview questions are in Appendix E.

Behavioral Analysis. As noted by Ma et al. [91], the primary data for behavior analysis in PBL sessions with ChatGPT comes from the interactions with the LLM counterpart. Thus, we exported all inquiry records from both the AutoPBL and Baseline sessions to Excel for behavioral analysis. Two researchers reviewed the inquiries of four participants together and generated codebooks based on inquiry content and motivation (see Appendix D). We treated the attempts at formulating a query during the prompt adjustment process as a single inquiry. Both researchers independently coded all 774 user inquiries, resolving any conflicts afterward. They also coded 719 checkpoint questions according to the taxonomy outlined in Figure 3, resolving conflicts collaboratively.

5.5.2 *Data Processing.*

The quiz was graded after the experiments. Each of the ten questions was worth 1 point. Participants received 0.5 points if they missed only one correct option in multiple-choice questions without selecting any incorrect options. Meanwhile, all questionnaire items were presented on a 7-point Likert scale, which we treated as continuous values for statistical analysis. All interviews were recorded via video conference. Transcripts were generated automatically, reviewed, edited to correct transcription errors, organized and translated by two researchers.

5.5.3 *Statistical Analysis.*

To assess knowledge-related learning outcomes, we analyzed relevant results (e.g., quiz scores, perceived gains, NASA TLX) from only the first session, as the two machine learning projects inevitably overlapped in some of the knowledge and concepts. For behavioral statistics, metacognition, and usability analyses, we combined data from both sessions to compare AutoPBL

with Baseline, as participants were evenly distributed across groups using a counterbalanced design.

We applied within-subjects statistical tests only when examining usability because we think it is unaffected by knowledge state changes. We applied between-subjects statistical tests for task load, behavioral analysis, and metacognition because we assume that, for these three aspects, the same user during different sessions cannot be considered the same subject due to changes in knowledge state and fatigue. We also applied between-subject tests to the learning outcomes because only first-session results were kept for analysis. Normality tests were applied to each group to determine whether to use parametric or non-parametric tests. A significance level of 0.05 was used for all analyses.

6 Results

6.1 Improving Learning Outcome

6.1.1 AutoPBL Effectively Improves Learning Outcomes by Emphasizing Key Learning Opportunities. In the user study, we used quiz performance after the first sessions to directly measure learning outcomes. As shown in Fig. 7(a), AutoPBL participants generally scored higher on both projects than those in the Baseline condition. Quiz scores on the four conditions followed normal distributions (Shapiro-Wilk normality test, $p = 0.09, 0.92, 0.72, 0.43$). So we conducted an unpaired t-test to compare between-subjects and found that AutoPBL users performed significantly better in both projects (*Spam Classification* $p = 0.02$, *Digit Recognition* $p = 0.03$).

We then analyzed the average score for each quiz question under the four conditions. As shown in Fig. 7(b), AutoPBL outperformed or tied with the Baseline condition on 8 out of 10 questions in both the *Spam Classification* and *Digit Recognition* projects. Notably, for three of the four questions where the Baseline condition scored higher, overall scoring rates were low (below 0.5 in both conditions), likely due to random variability. Overall, AutoPBL appears to help learners capitalize on learning opportunities that Baseline users may have missed.

Participants also shared insights that support our findings. Several noted that AutoPBL encouraged reflection on concepts that might be overlooked (P8, P9, P11, P13, P17). P9 observed that AutoPBL '*keeps reminding me to summarize previous knowledge points on my own and refine them*,' while P8 remarked that with Baseline, they felt '*the knowledge is mine, but it does not seem to have really sunk in.*' Some participants felt lost when navigating the lengthy document in the Baseline condition. P23 stated, '*You are presented with a large chunk of text all at once and expected to process it, which makes it hard to settle down and focus.*' In contrast, P23 described the AutoPBL experience as, '*It felt like progressing through levels in a game, defeating challenges along the way.*'

6.1.2 AutoPBL Helps Maintain a Balanced Focus on Execution, Learning, and Sensemaking. We also examined the subjective perceived gains in the SPBL process. The learning goal in SPBL can be divided into three components: conceptual comprehension, mastery of practical skills, and understanding of ML as a whole. In our study, participants self-reported their gains in each of these areas. As shown in Fig. 8(a), both projects demonstrated higher gains across all three aspects. However, the Mann-Whitney U test did not show statistical significance.

Evidence from user interviews revealed how AutoPBL helps learners maintain balance across the three aspects of learning goals. A common phenomenon observed in both previous [2, 4] and our studies is the users' tendency to focus on project completion (P13, P16, P26) as P13 described their learning process in the Baseline condition: '*I just skip over the concepts and theories and go straight to the goal. After looking at the goal, I skip the background information as well and go directly to the data loading part.*' In contrast, AutoPBL motivated participants to pay more attention to key knowledge areas (P11, P17).

Participants reported considerable improvements in their understanding of machine learning in both conditions (P20, P23, P24), likely because they were all novices before the study. However, some users acknowledged that the Baseline method did not help them understand the procedure deeply. As P15 explained: '*I feel confined by what is given to me. If I'm provided with a document like this, I just follow along unconsciously. To achieve something, I should think about the methods I can use, rather than just following step by step.*' Mindlessly following the tutorial is a common issue in SPBL, where comprehensive guidance limits opportunities for learners to critically engage with the process [4, 62]. AutoPBL addresses this by 1) keeping the overall framework visible to learners (P2, P3) and 2) encouraging reflection on the purpose of critical steps via checkpoint questions(P3, P26).

6.1.3 AutoPBL Fosters Success By Inducing More Efforts While Reducing Perceived Cognitive Burden. We investigated the perceived task load during the learning process using the NASA-TLX results from the first experimental session (Fig. 8(b)). Participants generally felt they performed well when using AutoPBL. The data also showed that AutoPBL significantly reduced frustration during SPBL (Mann-Whitney U test, $p = 0.01$) and caused less mental burden. As expected, participants needed more effort to complete SPBL tasks with AutoPBL. AutoPBL requires users to spend more time (average net time, removing tutorial block generation time in AutoPBL; AutoPBL 85 minutes, Baseline 58 minutes) and effort on understanding and reflecting on the learning process, which ultimately helps learners better achieve their goals (P9, P24). However, AutoPBL was perceived as less mentally demanding and exerted less time pressure. In the Baseline condition, users had to organize their learning process (P1, P12), craft prompts to get desired assistance (P26), and manage an overwhelming flow of information (P22, P23, P27). These efforts increased their mental load, leading to frustration and abandonment of learning goals. As P18 described the disorganized learning experience in the Baseline condition: '*It feels like I just skim through it without really thinking. But when it comes to actually doing it, I go back to look again, still don't understand it, and then just give up.*' In contrast, AutoPBL offered more beginner-friendly guidance (P2, P14). P2 remarked: '*You don't need any prior knowledge; it teaches you step by step from scratch. I think it's an excellent platform for learning from zero.*'

6.2 Changes in Behavior and Metacognition

6.2.1 AutoPBL Elicits More Positive Interactions with LLMs. Aside from learning outcomes, we conducted behavioral analysis through

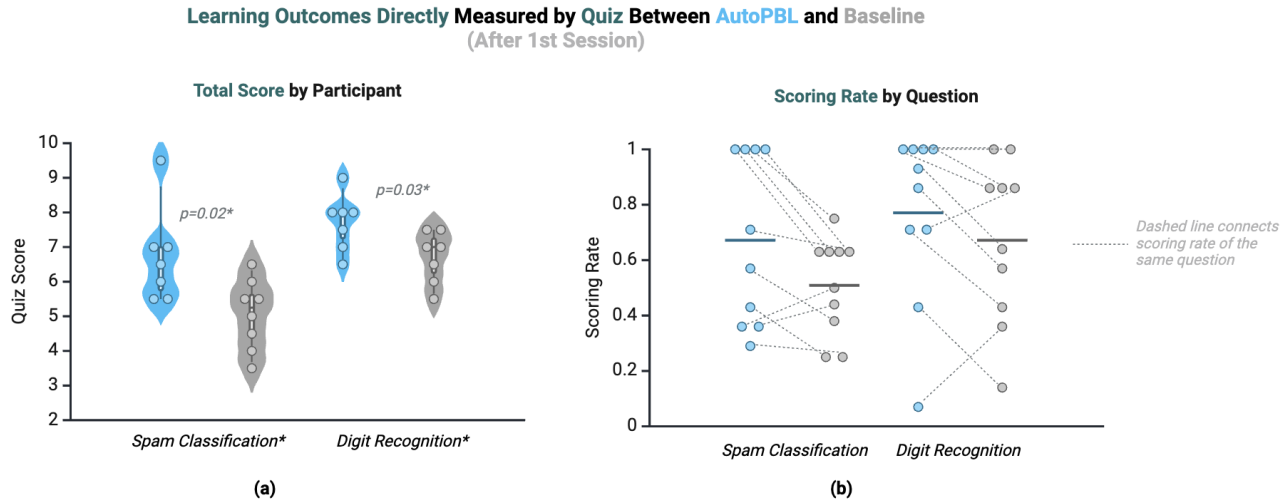


Figure 7: (a) To directly assess the effects of AutoPBL on learning outcomes, we used 10-question quizzes as probes. Since the two projects in the study are interrelated, we used the quiz results from each learner's first session. The violin plot illustrates the score distributions with 5-95 percentile whiskers. Participants using AutoPBL performed significantly better than those in the Baseline condition across both projects. (b) We also analyzed the average score for each question. A grey line connected data points for the same question. The AutoPBL group outperformed the Baseline group on most questions, indicating AutoPBL's effectiveness in helping learners grasp key learning opportunities during SPBL.

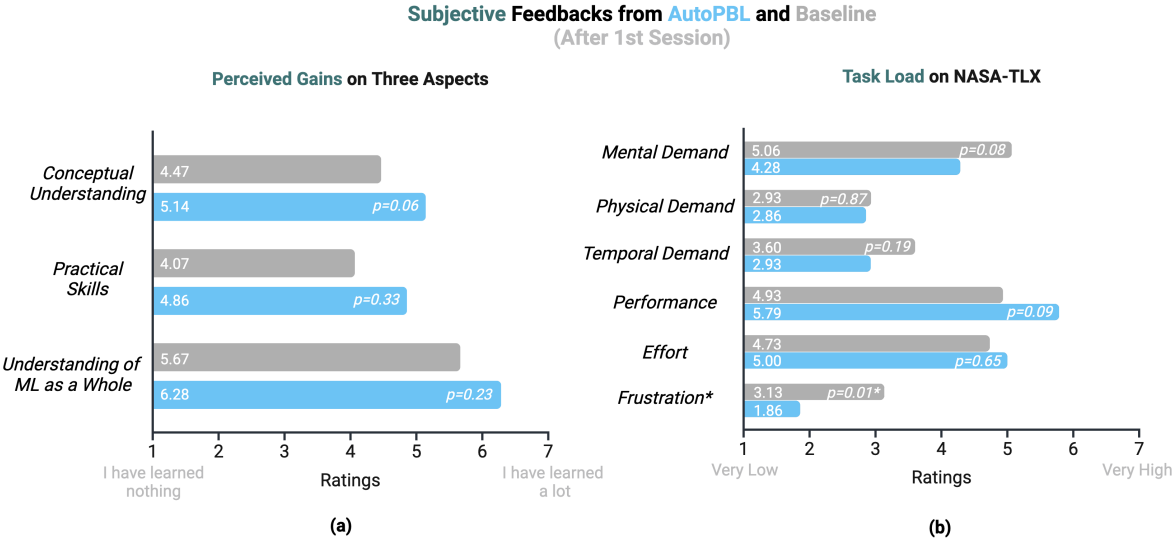


Figure 8: We examined the subjective aspects reported by users after their first experiment sessions. (a) The perceived gains were measured in three areas: conceptual comprehension, practical skills, and understanding of ML as a whole. AutoPBL users generally reported higher gains. (b) We also used the NASA-TLX scale after the first session to assess cognitive load. The results indicated that AutoPBL caused less frustration and mental demand, though it required comparable effort and physical demand during the SPBL process. Due to the non-normal distribution of many ratings, we applied the Mann-Whitney U test. Statistically significant ratings are marked with an asterisk (*).

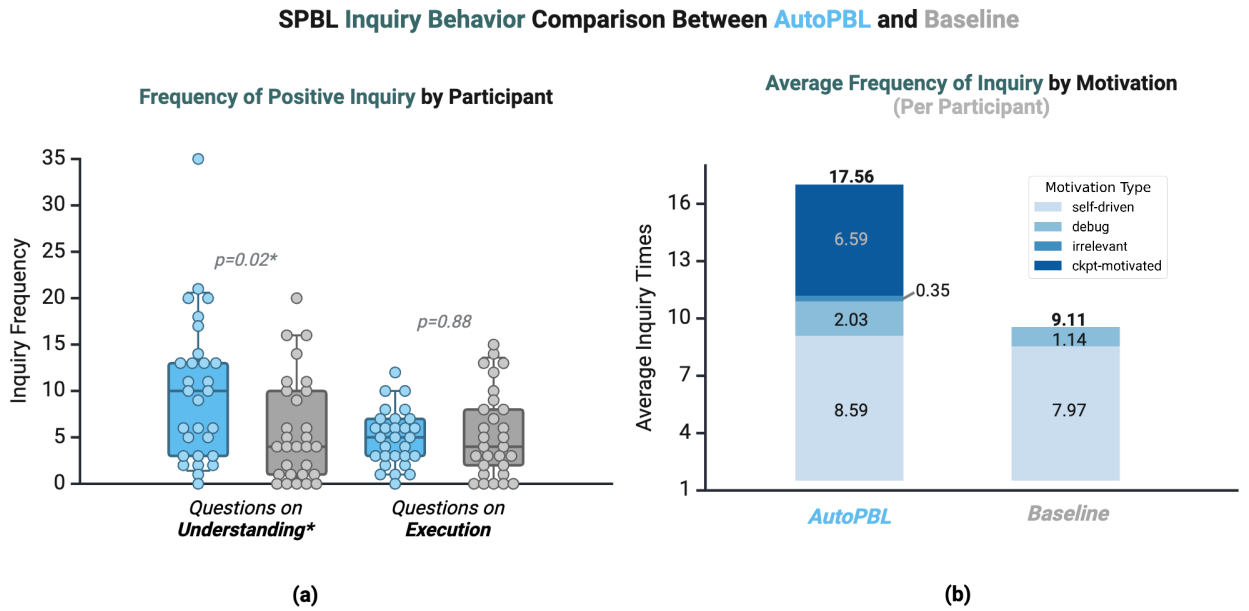


Figure 9: (a) We compared how often participants beneficially use LLM chatbots to accelerate understanding and task execution. Each data point represents a participant's inquiry behavior statistics across two topics. We found that user interacts significantly more with AI to fasten their understanding. (b) We visualized the average inquiry times across different motivation types. We found that AutoPBL promoted more inquiries by checkpoint questions. Statistically significant ratings are marked with an asterisk (*).

a comparison of how users query LLM chatbots. We first analyzed the frequency of beneficial inquiries, excluding irrelevant queries and queries aimed at obtaining direct answers to checkpoint questions. A Mann-Whitney U test was conducted to assess differences in inquiry behavior. As shown in Fig. 9(a), AutoPBL elicited significantly more questions to accelerate understanding. From the statistics on motivation types shown by Fig. 9 (b), we found that while participants asked a comparable number of self-driven questions, AutoPBL elicited more inquiries via checkpoint questions.

Participants noted that AutoPBL guided them toward key areas worth exploring, which motivated them to ask more questions (P1, P3, P16). After using both AutoPBL and the Baseline, P16 commented, '*If the tutorial doesn't ask me questions, I also don't know what questions I should ask the AI.*' The integration of the LLM chatbot into the tutorial interface and learning context also encouraged users to ask more questions by providing greater convenience (P2, P25). As P25 observed, '*Because it's a real-time interaction, asking questions becomes a bit more effortless. If it weren't for this interactive Q&A format during the learning process, I might have hesitated more about whether to ask a question.*'

6.2.2 AutoPBL Promotes High-quality Human-AI Collaboration in SPBL and Improves Self Confidence. Beyond the frequency of seeking assistance, we gathered metacognitive data on inquiry quality determined by interaction efficiency, question depth, and task allocation through self-ratings, as shown in Fig. 10. We found that participants felt they asked more proper and in-depth questions and collaborated with the AI significantly more effectively (Mann-Whitney U test, $p = 0.03$) when using AutoPBL.

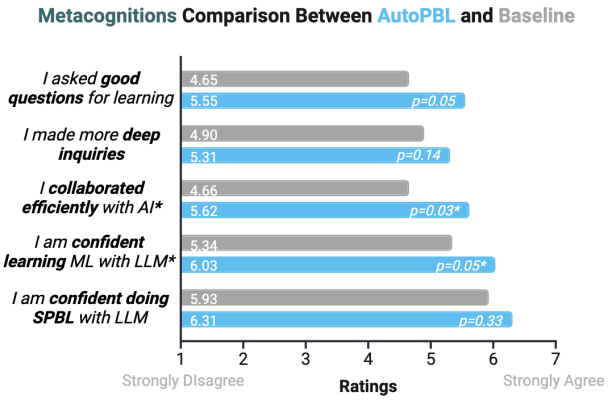


Figure 10: Through metacognitive questions, we found that AutoPBL enhanced the quality of human-AI collaboration in SPBL. Meanwhile, participants reported higher confidence in learning and SPBL with LLM after using AutoPBL. Statistically significant ratings are marked with an asterisk (*).

In addition to improving performance during the session, participants expressed that AutoPBL enhanced their ability to ask insightful questions (P2, P3) at the appropriate stages (P1, P9) during SPBL. As P1 explained, '*I think I can first ask about the general workflow, then break it down into smaller tasks and continue asking GPT. I've learned a strategy similar to what the platform provides.*' This sentiment is supported by a significant increase in confidence when learning ML with LLM (Mann-Whitney U test, $p = 0.05$) after using AutoPBL. However, no significant change in confidence was observed regarding self-directed project-based learning with the LLMs, possibly due to participants' expressed reliance on AutoPBL to guide the process (P5, P16).

6.3 AutoPBL's Usability in SPBL process

6.3.1 AutoPBL Brings Generally Favorable SPBL Experiences. To assess the overall user experience of AutoPBL compared to the Baseline, we asked users to rate both conditions on four aspects of usability: satisfaction, desire for future usage, ease of use, and pleasantness (Fig. 11(a)). In all four areas, our platform was generally preferred. We analyzed the ratings using a within-subject Wilcoxon-ranked test, as we assumed that session order had little impact on usability evaluation. The results showed that AutoPBL received significantly higher ratings for 'ease of use' than the Baseline. The significance aligns with our findings in Section 6.1.3, which suggest that AutoPBL aids workflow establishment and reduces frustration.

6.3.2 AutoPBL's Features are Generally Liked, With Room for Improvement. We further examined user feedback on AutoPBL's features by collecting participants' ratings (Fig. 11(b)). All features received an average rating above the neutral level (4), except for the 20-second countdown. The rating suggests that the design and implementation of most features achieved the desired effects. We analyzed participant feedback from interviews to better understand the variation in ratings. We drew three conclusions based on the analysis. Further discussions about the lessons learned from the results can be found in Section 7.1.

The structured block-based adaptive tutorial benefits beginners but should allow more flexibility. Participants valued features like the sidebar, block-by-block tutorials, and adaptive learning progress. These tools helped them stay organized (P6, P12, P20), engaged (P23), and under less pressure (P14). For example, P12 remarked on AutoPBL's adaptive generation: 'You can always know if you have absorbed the knowledge. If you haven't, it will tell you where to improve.' However, some wanted more freedom, such as skipping complicated steps or content they were not interested in (P1, P15). P1 noted, '*Some steps you already know or don't want to waste time on, and you should be able to skip them.*' The countdown feature, intended to encourage engagement on tutorial content, was criticized for not matching the actual reading time needed (P9, P15) and for preventing goal-setting by looking ahead at checkpoint questions (P27). Additionally, some participants preferred to read the tutorial fully to grasp the process and challenges (P1, P4, P11, P13).

The virtual TA is easy to use, but non-in-context inquiries should be allowed. The virtual TA received the highest rating due to its in-context question answering and seamless integration. The quote-and-ask function and preset question modes made it easier for users to express their needs and the context of the inquiry and receive relevant responses, boosting confidence in asking questions (P7, P11, P12). However, some users failed to capture the difference brought by selecting a preset question mode, leading to confusion (P9, P21). Without switching between the chatbot and tutorial windows, users felt more engaged and willing to ask questions (P6, P10, P25). However, some participants noted drawbacks of the in-context Q&A (P7, P13). P13 shared, '*I wanted to work on each piece of code one by one, but it gave me everything at once.*'

Appropriate checkpoint questions should appear in the right circumstance. Participants praised the design of these questions for highlighting key learning opportunities (See 6.1.1). Participants also noted that checkpoints and adaptive responses helped them confirm their understanding or correct mistakes promptly (P9, P12, P19). However, feature ratings indicated some dissatisfaction. Users explained that low ratings stemmed from poorly timed, which disrupted their learning (P6, P14, P15, P16).

We examined the effectiveness and usability of specific types of checkpoint questions, as outlined in Fig. 3. Participants rated the effectiveness and usability of each checkpoint they encountered. As shown in Fig. 12, checkpoint questions on *How and Why Things Work*, *Application of Knowledge*, and *Description of Observation* were generally adequate, helping users reflect on newly learned concepts or recent practice. However, the effectiveness of *Basic Information Retrieval* and *Code Completion and Explanation* was debated. Some participants felt these questions were sufficient to inform them of the key points (P6, P17), while others preferred more challenging, thought-provoking questions like *Application of Knowledge* (P14, P19). As P14 remarked: "I prefer questions that are a bit more open-ended, ones that encourage me to think, rather than simply copying or imitating." *Code Completion* faced criticism for low usability (Fig. 12(b)), as participants who were new to machine learning found it too challenging (P11, P15). In contrast, some felt it did not adequately help them develop programming skills (P4, P14, P21). Checkpoints on *Real-world Application* and *Exploration* received limited praise, possibly because participants in a paid user study setting did not need as much external motivation. *Progress Checkpoints* were rated least effective, as they required minimal thought, which was expected since they were designed to track learner progress rather than provide direct guidance.

7 Discussion

7.1 Improving LLM-Generated Teaching Content

All teaching materials, including tutorials for AutoPBL and Baseline, checkpoint questions, and responses to user inquiries, were generated using OpenAI's GPT-4o API. While content quality was generally acceptable, participants raised concerns about the generated content's trustworthiness, teaching methods, and adaptability. These issues are common in LLM-generated educational content [12], and we propose solutions.

Building Trust through Quality Content. As we discovered in the technical evaluation in 5.1.2 and during the experiment, GPT-4 occasionally made mistakes in generating checkpoint questions and evaluating user responses. These errors disrupted the learning experience (P6, P27). Meanwhile, participants with prior

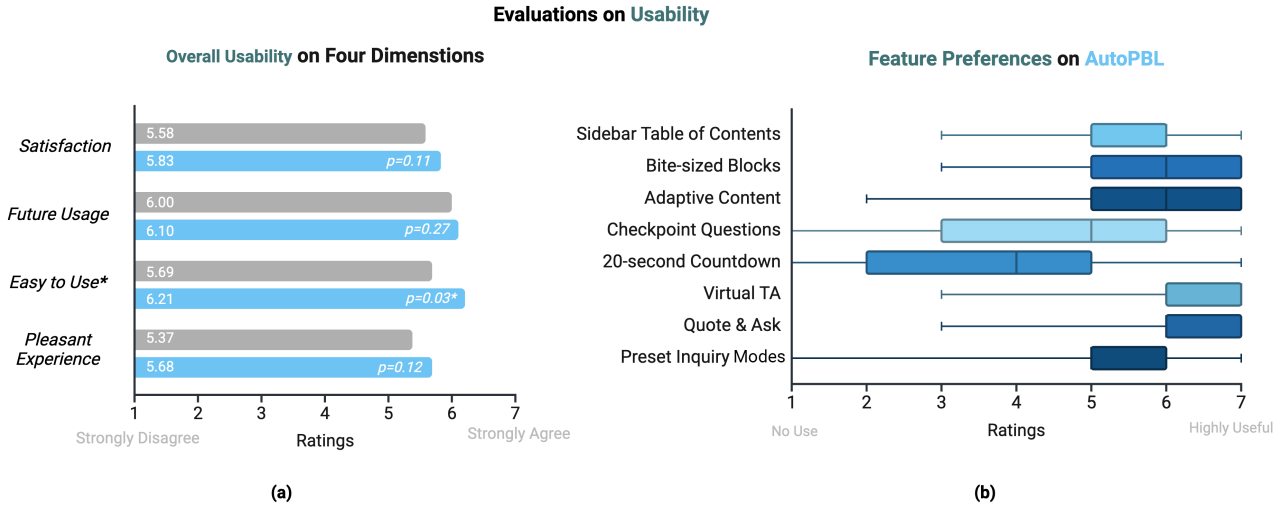


Figure 11: (a) We found that AutoPBL was generally preferred for usability on all four aspects. To test the significance of this preference, we conducted a Wilcoxon-ranked test and marked the statistically significant difference with an asterisk (*). (b) To understand how participants perceived AutoPBL's features, we asked them to rate their overall preference for each. The results were visualized using a boxplot, which includes the mean value and a 5-95 percentile range.

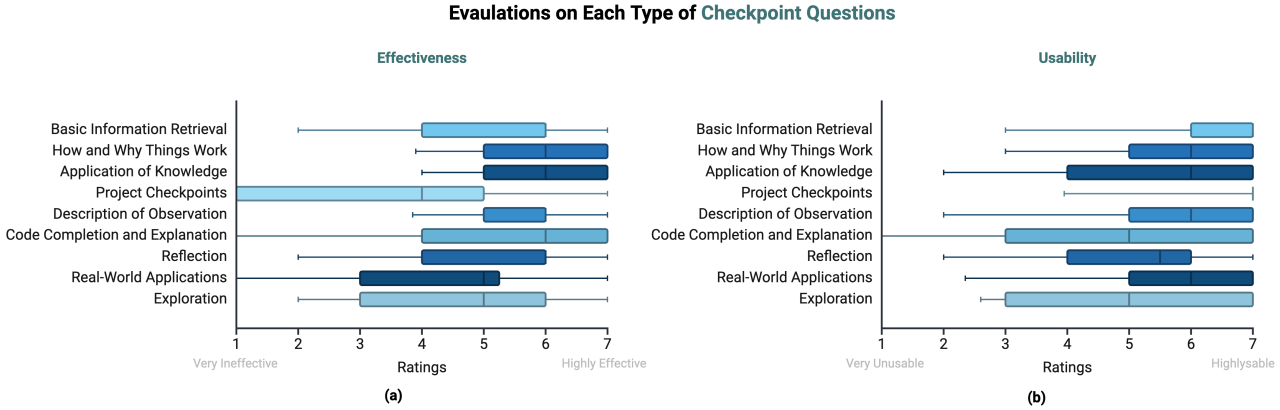


Figure 12: We asked participants to evaluate the effectiveness and usability of all the checkpoint questions they received. The questions were categorized based on Fig. 3. The results for each category are presented using boxplots with 5-95 percentile ranges.

knowledge of LLM hallucinations (P5, P7) voiced concerns about trusting LLM-generated content. In response, we plan to integrate human-reviewed content into future versions of AutoPBL. Possible solutions include using Retrieval-Augmented Generation (RAG) to reduce hallucinations [40]. Moreover, references should be directly displayed in tutorial content and framework to improve trust in future versions of AutoPBL.

Improving Content Organization. While LLMs can generate accurate and readable tutoring content [12], participants suggested improvements to make tutorials more intuitive. These included reorganizing content, providing examples at the right points, and offering visual aids based on topic complexity (P9, P15). We believe these enhancements can be achieved through RAG and more advanced multi-agent system designs.

Adapting Teaching Methods to Learners' Knowledge States and Habits. We also realized the need to tailor content to users' knowledge states and learning habits. Some participants preferred shorter content (P29) and iterative responses to their inquiries (P7, P13). However, LLMs tend to offer comprehensive answers [42], which may perform well on productivity-oriented benchmarks but do not always suit individual learning habits. Some users also reported that the checkpoint questions were too easy or difficult

(P15, P19). We recognize the importance of explicitly modeling and tracking mastery, a key feature of traditional intelligent tutoring systems [1, 56]. Participants also expressed differing preferences for checkpoint question types (see Section 6.3.1), leading us to consider personalizing content based on self-reported or inferred learning habits.

Crowdsourcing Evaluation of Teaching Content. Tailored teaching is a central goal in education [46]. Achieving this requires understanding what, when, and how to teach a specific learner. AutoPBL shows promise in gathering user feedback on teaching content. By combining this feedback with users' knowledge state, learning style, and context, we could possibly refine adaptive content generation through methods like fine-tuning.

In conclusion, our study identified challenges with LLM-generated content in the AutoPBL system and proposed solutions. These findings are not limited to AutoPBL but apply broadly to LLM-supported learning. We urge the community to focus on creating AI-generated content that better suits the learning process.

7.2 Beyond SPBL on Machine Learning

AutoPBL explores LLM-embedded PBL platforms with a focus on SPBL and the topic of machine learning. SPBL lacks specific components valued in traditional PBL, such as self-direction and team collaboration, due to the nature of self-learning and reliance on guiding materials [66, 82]. We chose machine learning as an example because it allows for convenient progress synchronization. This section discusses the potential for extending AutoPBL to a broader PBL context, specifically regarding diverse learning topics, self-direction, and team collaboration.

Adapting AutoPBL to More PBL Topics. AutoPBL is not limited to supporting machine learning or computer science projects (see Section 4.4). As long as LLM is capable in the domain and the project progress can be communicated to the system (through text or other format), AutoPBL can potentially work. Participants from mechanical and environmental engineering departments expressed interest in using AutoPBL in their fields (P12, P29, P22). Some participants also suggested integrating Jupyter Notebook into AutoPBL's interface and content generation (P19, P27), which we believe is feasible. Similar integration can benefit other SPBL topics if the project's progress can be effectively synchronized with the LLM agents.

Enabling Self-Direction on AutoPBL. A key feature of AutoPBL is its ability to dynamically generate content for the next block, allowing for a flexible project path. In our user study, we only kept the tutorial framework static to ensure equal learning materials for all participants. However, to enable true self-direction, we need features that allow users to modify their project paths on the sidebar framework and adjust the direction of their next steps after completing each block.

Supporting Team Collaboration on AutoPBL. Initially, we did not design AutoPBL for collaborative settings. However, one participant (P13) noted the potential for team members to use the platform together for project work. P13 envisioned a scenario where '*when others are responsible for writing certain functions, and you need to use those functions, collaboration becomes necessary. AI must not only relate to the tutorial content but also include content written by others. The AI can help complete a large project by referencing teammates' work.*' Such a feature can enable participants to identify opportunities for team communication. P13 also suggested improving the current tutorial framework by using a tree-view structure. This design would represent each member's tasks as nodes and subtrees, allowing the team to track task allocation, individual progress, and the timing for integrating their work.

In summary, we used empirical findings on AutoPBL as a lens to consider its broader applicability in PBL. We identified several areas for improvement, including supporting more topics, enabling self-direction, and facilitating team collaboration, which will be addressed in future work.

7.3 Limitations

First, we evaluated AutoPBL specifically in the context of machine learning. While we demonstrated its effectiveness in ML, further work is needed to confirm whether these results generalize to other SPBL domains, such as programming language learning, software usage, surgery operations, or cooking. Different learning goals may require adjustments to AutoPBL's features. Although the implementation framework is designed to be generalizable, we have not yet systematically examined AutoPBL's performance in areas where LLM knowledge is limited. If the generated content is subpar, we remain uncertain whether simple RAG methods would sufficiently enhance it.

Second, we restricted the participants to novice learners to facilitate a fair comparison of learning outcomes. However, in real-world SPBL, learners may have varying levels of prior knowledge. For instance, a computer science professional might use SPBL to learn a new programming language. We did not examine whether AutoPBL's effectiveness for beginners can be generalized to learners with higher prior expertise.

Third, we focused primarily on participants' inquiry behaviors for behavioral analysis. While other behaviors could be studied, they are challenging to capture. For example, in pilot studies, we attempted to use a think-aloud protocol [18] but found that speaking interfered with the learning process, and users typically forgot to talk. We also tried analyzing screen recordings but found that the behaviors captured were too ambiguous for us to examine. For example, it was difficult to distinguish whether participants were carefully reading a tutorial or distracted. To further investigate AutoPBL's behavioral impact, we need to collect additional information, such as eye-tracking [27] or other physiological data [28], without disrupting the learning process.

Additionally, due to time constraints and the long timespan required for SPBL, we conducted only a lab study with 29 participants. As a result, some of our quantitative evaluation results were not statistically significant. We expect more robust results with more participants. To this end, we plan to deploy AutoPBL across multiple university courses in the following semester to gather more empirical data. Additionally, we aim to test for changes in metacognitive skills through extended learning sessions across various topics. We will also focus on shifts in internal abilities, such as creativity, critical thinking, and problem-solving, which previous research suggests are also vital benefits of traditional PBL [22].

Finally, AutoPBL relies on relatively heavy GPT-4o API usage. In our study, the cost per learning session generally falls in the range of $2 to $3. While this is not overly prohibitive, we are actively seeking ways to systematically reduce the LLM usage of AutoPBL to lower cost and waiting time.

8 Conclusion

We introduced AutoPBL, a self project-based learning platform that provides guidance and support with systematic and responsible integration of LLM. By clarifying current priorities and providing timely assistance, AutoPBL improves learning outcomes and elicits better learning behaviors and metacognition. A user study in which 29 beginners learned through ML projects shows that our platform makes self-PBL more effective. AutoPBL helps learners stay engaged, organized, and motivated during the learning process. We hope our work will inspire future research and development of LLM-integrated project-based learning tools and platforms that can make effective PBL ubiquitous.

Acknowledgments

This work is supported by the Natural Science Foundation of China under Grant No. 62132010, Beijing Key Lab of Networked Multimedia, Institute for Artificial Intelligence, Tsinghua University (THUAI), Beijing National Research Center for Information Science and Technology (BNRist), 2025 Key Technological Innovation Program of Ningbo City under Grant No. 2022Z080, Beijing Municipal Science and Technology Commission, Administrative Commission of Zhongguancun Science Park No.Z221100006722018, and Science and Technology Innovation Key R&D Program of Chongqing.

References

[1] Ghodai Abdelrahman, Qing Wang, and Bernardo Nunes. 2023. Knowledge tracing: A survey. *Comput. Surveys* 55, 11 (2023), 1–37.

[2] Shaban Aldabbus. 2018. Project-based learning: Implementation & challenges. *International journal of education, learning and development* 6, 3 (2018), 71–79.

[3] Tiago Almeida and Jos Hidalgo. 2011. SMS Spam Collection. UCI Machine Learning Repository. DOI: https://doi.org/10.24432/C5CC84.

[4] Kai Arakawa, Qiang Hao, Tyler Greer, Lu Ding, Christopher D Hundhausen, and Abigayle Peterson. 2021. In situ identification of student self-regulated learning struggles in programming assignments. In *Proceedings of the 52nd ACM Technical Symposium on Computer Science Education*. 467–473.

[5] Paul Black and Dylan Wiliam. 2009. Developing the theory of formative assessment. *Educational Assessment, Evaluation and Accountability (formerly: Journal of personnel evaluation in education)* 21 (2009), 5–31.

[6] Phyllis C Blumenfeld, Elliot Soloway, Ronald W Marx, Joseph S Krajcik, Mark Guzdial, and Annemarie Palincsar. 1991. Motivating project-based learning: Sustaining the doing, supporting the learning. *Educational psychologist* 26, 3-4 (1991), 369–398.

[7] Dan Carpenter, Wookhee Min, Seung Lee, Gamze Ozogul, Xiaoying Zheng, and James Lester. 2024. Assessing student explanations with large language models using fine-tuning and few-shot learning. In *Proceedings of the 19th Workshop on Innovative Use of NLP for Building Educational Applications (BEA 2024)*. 403–413.

[8] David Cerna, Martina Seidl, Wolfgang Schreiner, Wolfgang Windsteiger, and Armin Biere. 2020. Aiding an Introduction to Formal Reasoning Within a First-Year Logic Course for CS Majors Using a Mobile Self-Study App. In *Proceedings of the 2020 ACM Conference on Innovation and Technology in Computer Science Education*. 61–67.

[9] Kathy Charmaz. 2006. *Constructing grounded theory: A practical guide through qualitative analysis*. Sage.

[10] Sankalan Pal Chowdhury, Vilém Zouhar, and Mrinmaya Sachan. 2024. AutoTutor meets Large Language Models: A Language Model Tutor with Rich Pedagogy and Guardrails. In *ACM Conference on Learning @ Scale*. https://api.semanticscholar.org/CorpusID:267657803

[11] Albert T. Corbett and John R. Anderson. 2005. Knowledge tracing: Modeling the acquisition of procedural knowledge. *User Modeling and User-Adapted Interaction* 4 (2005), 253–278. https://api.semanticscholar.org/CorpusID:19228797

[12] Wei Dai, Jionghao Lin, Hua Jin, Tongguang Li, Yi-Shan Tsai, Dragan Gašević, and Guanliang Chen. 2023. Can large language models provide feedback to students? A case study on ChatGPT. In *2023 IEEE International Conference on Advanced Learning Technologies (ICALT)*. IEEE, 323–325.

[13] Hai Dang, Lukas Mecke, Florian Lehmann, Sven Goller, and Daniel Buschek. 2022. How to prompt? Opportunities and challenges of zero-and few-shot learning for human-AI interaction in creative applications of generative models. *arXiv preprint arXiv:2209.01390* (2022).

[14] Yossi Ben David, Avi Segal, and Ya'akov Gal. 2016. Sequencing educational content in classrooms using Bayesian knowledge tracing. In *Proceedings of the sixth international conference on Learning Analytics & Knowledge*. 354–363.

[15] Jimmy De La Torre. 2009. DINA Model and Parameter Estimation: A Didactic. *Journal of Educational and Behavioral Statistics* 34, 1 (March 2009), 115–130. doi:10.3102/1076998607309474

[16] Li Deng. 2012. The mnist database of handwritten digit images for machine learning research. *IEEE Signal Processing Magazine* 29, 6 (2012), 141–142.

[17] Yuyang Ding, Hanglei Hu, Jie Zhou, Qin Chen, Bo Jiang, and Liang He. 2024. Boosting Large Language Models with Socratic Method for Conversational Mathematics Teaching. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management* (Boise, ID, USA) *(CIKM '24)*. Association for Computing Machinery, New York, NY, USA, 3730–3735. doi:10.1145/3627673.3679881

[18] Rachel J. Ebner and Linnea C. Ehri. 2013. Vocabulary Learning on the Internet: Using a Structured Think-Aloud Procedure. *Journal of Adolescent & Adult Literacy* 56, 6 (March 2013), 480–489. doi:10.1002/JAAL.169

[19] Abdellah Ibrahim Mohammed Elfeky and Marwa Yasien Helmy Elbyaly. 2023. The Effect Of E-Tutorial Programs On Improving The Producing Digital Content Skill. *European Chemical Bulletin* 12 (2023), 6581–6587.

[20] Mary C English and Anastasia Kitsantas. 2013. Supporting student self-regulated learning in problem-and project-based learning. *Interdisciplinary journal of problem-based learning* 7, 2 (2013), 6.

[21] Melissa Sommerfeld Gresalfi, Jacqueline Barnes, and Dionne Cross. 2012. When does an opportunity become an opportunity? Unpacking classroom practice through the lens of ecological psychology. *Educational Studies in Mathematics* 80 (2012), 249–267.

[22] Pengyue Guo, Nadira Saab, Lysanne S Post, and Wilfried Admiraal. 2020. A review of project-based learning in higher education: Student outcomes and measures. *International journal of educational research* 102 (2020), 101586.

[23] Saad Hassan, Sooyeon Lee, Dimitris Metaxas, Carol Neidle, and Matt Huenerfauth. 2022. Understanding ASL Learners' Preferences for a Sign Language Recording and Automatic Feedback System to Support Self-Study. In *Proceedings of the 24th International ACM SIGACCESS Conference on Computers and Accessibility*. 1–5.

[24] Lei Huang. 2019. Integrating machine learning to undergraduate engineering curricula through project-based learning. In *2019 IEEE Frontiers in Education Conference (FIE)*. IEEE, 1–4.

[25] Lei Huang, Weijiang Yu, Weitao Ma, Weihong Zhong, Zhangyin Feng, Haotian Wang, Qianglong Chen, Weihua Peng, Xiaocheng Feng, Bing Qin, and Ting Liu. 2024. A Survey on Hallucination in Large Language Models: Principles, Taxonomy, Challenges, and Open Questions. *ACM Transactions on Information Systems* (Nov. 2024). doi:10.1145/3703155

[26] Zhenya Huang, Qi Liu, Chengxiang Zhai, Yu Yin, Enhong Chen, Weibo Gao, and Guoping Hu. 2019. Exploring multi-objective exercise recommendations in online education systems. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*. 1261–1270.

[27] Halszka Jarodzka, Irene Skuballa, and Hans Gruber. 2021. Eye-Tracking in Educational Practice: Investigating Visual Perception Underlying Teaching and Learning in the Classroom. *Educational Psychology Review* 33, 1 (March 2021), 1–10. doi:10.1007/s10648-020-09565-7

[28] Norbert Jaušovec and Ksenija Jaušovec. 2000. EEG activity during the performance of complex mental problems. *International Journal of Psychophysiology* 36, 1 (2000), 73–88.

[29] Jaeho Jeon and Seongyong Lee. 2023. Large language models in education: A focus on the complementary relationship between human teachers and ChatGPT. *Education and Information Technologies* 28, 12 (2023), 15873–15892.

[30] Ziwei Ji, Tiezheng Yu, Yan Xu, Nayeon Lee, Etsuko Ishii, and Pascale Fung. 2023. Towards mitigating LLM hallucination via self reflection. In *Findings of the Association for Computational Linguistics: EMNLP 2023*. 1827–1843.

[31] Hyoungwook Jin, Seonghee Lee, Hyungyu Shin, and Juho Kim. 2024. Teach AI How to Code: Using Large Language Models as Teachable Agents for Programming Education. In *Proceedings of the CHI Conference on Human Factors in Computing Systems* (Honolulu, HI, USA) *(CHI '24)*. Association for Computing Machinery, New York, NY, USA, Article 652, 28 pages. doi:10.1145/3613904.3642349

[32] Enkelejda Kasneci, Kathrin Seßler, Stefan Küchemann, Maria Bannert, Daryna Dementieva, Frank Fischer, Urs Gasser, Georg Groh, Stephan Günnemann, Eyke Hüllermeier, et al. 2023. ChatGPT for good? On opportunities and challenges of large language models for education. *Learning and individual differences* 103 (2023), 102274.

[33] Majeed Kazemitabaar, Runlong Ye, Xiaoning Wang, Austin Zachary Henley, Paul Denny, Michelle Craig, and Tovi Grossman. 2024. Codeaid: Evaluating a classroom deployment of an llm-based programming assistant that balances student and educator needs. In *Proceedings of the CHI Conference on Human Factors in Computing Systems*. 1–20.

[34] Dimitra Kokotsaki, Victoria Menzies, and Andy Wiggins. 2016. Project-based learning: A review of the literature. *Improving schools* 19, 3 (2016), 267–277.

[35] Joseph S Krajcik and Phyllis C Blumenfeld. 2006. *Project-based learning*. na.

[36] Joseph S Krajcik, Charlene Lochbihler Czerniak, and Carl F Berger. 2003. *Teaching science in elementary and middle school classrooms: A project-based approach*. McGraw-Hill Humanities, Social Sciences & World Languages.

[37] Harsh Kumar, Ilya Musabirov, Mohi Reza, Jiakai Shi, Anastasia Kuzminykh, Joseph Jay Williams, and Michael Liut. 2023. Impact of guidance and interaction strategies for LLM use on Learner Performance and perception. *arXiv preprint arXiv:2310.13712* (2023).

[38] Tiffany H Kung, Morgan Cheatham, Arielle Medenilla, Czarina Sillos, Lorie De Leon, Camille Elepaño, Maria Madriaga, Rimel Aggabao, Giezel Diaz-Candido, James Maningo, et al. 2023. Performance of ChatGPT on USMLE: potential for AI-assisted medical education using large language models. *PLoS digital health* 2, 2 (2023), e0000198.

[39] James Larson, Shawn S. Jordan, Micah Lande, and Steven Weiner. 2020. Supporting Self-Directed Learning in a Project-Based Embedded Systems Design Course. *IEEE Transactions on Education* 63, 2 (2020), 88–97. doi:10.1109/TE.2020.2975358

[40] Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in Neural Information Processing Systems* 33 (2020), 9459–9474.

[41] Hengyu Liu, Tiancheng Zhang, Fan Li, Minghe Yu, and Ge Yu. 2024. A probabilistic generative model for tracking multi-knowledge concept mastery probability. *Frontiers of Computer Science* 18, 3 (2024), 183602.

[42] Mengqi Liu and Faten M'hiri. 2024. Beyond Traditional Teaching: Large Language Models as Simulated Teaching Assistants in Computer Science. In *Proceedings of the 55th ACM Technical Symposium on Computer Science Education V. 1*. 743–749.

[43] Peiyan Liu. [n. d.]. Optimizing Self-Paced Learning in Machine Learning Education for Working Professionals: Strategies, Trends, and Insights. ([n. d.]).

[44] Zhengyuan Liu, Stella Xin Yin, Geyu Lin, and Nancy F. Chen. 2024. Personality-aware Student Simulation for Conversational Intelligent Tutoring Systems. arXiv:2404.06762 [cs.CL] https://arxiv.org/abs/2404.06762

[45] Taiming Lu, Muhan Gao, Kuai Yu, Adam Byerly, and Daniel Khashabi. 2024. Insights into LLM Long-Context Failures: When Transformers Know but Don't Tell. arXiv:2406.14673 [cs.CL] https://arxiv.org/abs/2406.14673

[46] Shuai Ma, Taichang Zhou, Fei Nie, and Xiaojuan Ma. 2022. Glancee: An Adaptable System for Instructors to Grasp Student Learning Status in Synchronous Online Classes *(CHI '22)*. Association for Computing Machinery, New York, NY, USA, Article 313, 25 pages. doi:10.1145/3491102.3517482

[47] Jakub Macina, Nico Daheim, Sankalan Pal Chowdhury, Tanmay Sinha, Manu Kapur, Iryna Gurevych, and Mrinmaya Sachan. 2023. MathDial: A Dialogue Tutoring Dataset with Rich Pedagogical Properties Grounded in Math Reasoning Problems. arXiv:2305.14536 [cs.CL] https://arxiv.org/abs/2305.14536

[48] Terry Mayes and Sara De Freitas. 2007. Learning and e-learning: the role of theory. In *Rethinking pedagogy for a digital age*. Routledge, 33–45.

[49] Nanxi Meng, Yan Dong, Dorian Roehrs, and Lin Luan. 2023. Tackle implementation challenges in project-based learning: a survey study of PBL e-learning platforms. *Educational technology research and development* 71, 3 (2023), 1179–1207.

[50] Jesse G Meyer, Ryan J Urbanowicz, Patrick CN Martin, Karen O'Connor, Ruowang Li, Pei-Chen Peng, Tiffani J Bright, Nicholas Tatonetti, Kyoung Jae Won, Graciela Gonzalez-Hernandez, et al. 2023. ChatGPT and large language models in academia: opportunities and challenges. *BioData Mining* 16, 1 (2023), 20.

[51] Fatma Miladi, Valéry Psyché, and Daniel Lemire. 2024. Leveraging GPT-4 for Accuracy in Education: A Comparative Study on Retrieval-Augmented Generation in MOOCs. In *International Conference on Artificial Intelligence in Education*. Springer, 427–434.

[52] George A Miller. 1956. The magical number seven, plus or minus two: Some limits on our capacity for processing information. *Psychological review* 63, 2 (1956), 81.

[53] Brent Mittelstadt, Sandra Wachter, and Chris Russell. 2023. To protect science, we must use LLMs as zero-shot translators. *Nature Human Behaviour* 7, 11 (2023), 1830–1832.

[54] Gona Sirwan Mohammed, Karzan Wakil, and Sarkhell Sirwan Nawroly. 2018. The effectiveness of microlearning to improve students' learning ability. *International Journal of Educational Research Review* 3, 3 (2018), 32–38.

[55] Somayyeh Mousavian and Hossein Siahpoosh. 2018. The Effects of vocabulary pre-teaching and pre-questioning on intermediate Iranian EFL learners' reading comprehenstion ability. *International Journal of Applied Linguistics and English Literature* 7, 2 (2018), 58–63.

[56] Elham Mousavinasab, Nahid Zarifsanaiey, Sharareh R. Niakan Kalhori, Mahnaz Rakhshan, Leila Keikha, and Marjan Ghazi Saeedi. 2021. Intelligent tutoring systems: a systematic review of characteristics, applications, and evaluation methods. *Interactive Learning Environments* 29, 1 (2021), 142–163.

[57] Srishti Palani, Zijian Ding, Stephen MacNeil, and Steven P. Dow. 2021. The "Active Search" Hypothesis: How Search Strategies Relate to Creative Learning. In *Proceedings of the 2021 Conference on Human Information Interaction and Retrieval* (Canberra ACT, Australia) *(CHIIR '21)*. Association for Computing Machinery, New York, NY, USA, 325–329. doi:10.1145/3406522.3446046

[58] Zachary A Pardos, Matthew Tang, Ioannis Anastasopoulos, Shreya K Sheel, and Ethan Zhang. 2023. Oatutor: An open-source adaptive tutoring system and curated content library for learning sciences research. In *Proceedings of the 2023 chi conference on human factors in computing systems*. 1–17.

[59] Minju Park, Sojung Kim, Seunghyun Lee, Soonwoo Kwon, and Kyuseok Kim. 2024. Empowering personalized learning through a conversation-based tutoring system with student modeling. In *Extended Abstracts of the CHI Conference on Human Factors in Computing Systems*. 1–10.

[60] Jun Peng, Minhong Wang, and Demetrios Sampson. 2017. Scaffolding project-based learning of computer programming in an online learning environment. In *2017 IEEE 17th International Conference on Advanced Learning Technologies (ICALT)*. IEEE, 315–319.

[61] Paul R Pintrich. 2004. A conceptual framework for assessing motivation and self-regulated learning in college students. *Educational psychology review* 16 (2004), 385–407.

[62] Prajish Prasad and Aamod Sane. 2024. A Self-Regulated Learning Framework using Generative AI and its Application in CS Educational Intervention Design. In *Proceedings of the 55th ACM Technical Symposium on Computer Science Education V. 1*. 1070–1076.

[63] Md Mamunur Rashid, Nilsu Atilgan, Jonathan Dobres, Stephanie Day, Veronika Penkova, Mert Küçük, Steven R Clapp, and Ben D Sawyer. 2023. Humanizing AI in Education: A Readability Comparison of LLM and Human-Created Educational Content. In *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*. SAGE Publications Sage CA: Los Angeles, CA, 10711811241261689.

[64] Chantal Roddy, Danielle Lalaine Amiet, Jennifer Chung, Christopher Holt, Lauren Shaw, Stephen McKenzie, Filia Garivaldis, Jason M Lodge, and Matthew Edward Mundy. 2017. Applying best practice online learning, teaching, and support to intensive online environments: An integrative review. In *Frontiers in Education*, Vol. 2. Frontiers Media SA, 59.

[65] Phil Sands and Aman Yadav. 2020. Self-regulation for high school learners in a MOOC computer science course. In *Proceedings of the 51st ACM technical symposium on computer science education*. 845–851.

[66] Willy Johan Saputra, Riyan Leandros, Reagen Yohanes Sayoga, and Dina Fitria Murad. 2022. Development of Transparent Rubric System Design As A Standardization of Assessments For Lecturer Effectiveness And Encourage Students In Self-Study. In *Proceedings of the 7th International Conference on Sustainable Information Engineering and Technology*. 247–252.

[67] Robin Schmucker, Meng Xia, Amos Azaria, and Tom Mitchell. 2024. Ruffle&Riley: From Lesson Text to Conversational Tutoring. In *Proceedings of the Eleventh ACM Conference on Learning@ Scale*. 547–549.

[68] Yiyin Shen, Xinyi Ai, Adalbert Gerald Soosai Raj, Rogers Jeffrey Leo John, and Meenakshi Syamkumar. 2024. Implications of ChatGPT for Data Science Education. In *Proceedings of the 55th ACM Technical Symposium on Computer Science Education V. 1*. 1230–1236.

[69] Myeong-Hee Shin. 2018. Effects of Project-Based Learning on Students' Motivation and Self-Efficacy. *English teaching* 73, 1 (2018), 95–114.

[70] Leonardo Silva, António Mendes, Anabela Gomes, and Gabriel Fortes. 2024. What Learning Strategies are Used by Programming Students? A Qualitative Study Grounded on the Self-regulation of Learning Theory. *ACM Transactions on Computing Education* 24, 1 (2024), 1–26.

[71] John Stamper, Ruiwei Xiao, and Xinying Hou. 2024. Enhancing LLM-Based Feedback: Insights from Intelligent Tutoring Systems and the Learning Sciences. In *Artificial Intelligence in Education. Posters and Late Breaking Results, Workshops and Tutorials, Industry and Innovation Tracks, Practitioners, Doctoral Consortium and Blue Sky*, Andrew M. Olney, Irene-Angelica Chounta, Zitao Liu, Olga C. Santos, and Ig Ibert Bittencourt (Eds.). Springer Nature Switzerland, Cham, 32–43.

[72] Candice Stefanou, Jonathan D Stolk, Michael Prince, John C Chen, and Susan M Lord. 2013. Self-regulation and autonomy in problem-and project-based learning environments. *Active Learning in Higher Education* 14, 2 (2013), 109–122.

[73] John Sweller. 1988. Cognitive load during problem solving: Effects on learning. *Cognitive science* 12, 2 (1988), 257–285.

[74] Ai-dung Taylor and Woei Hung. 2022. The effects of microlearning: A scoping review. *Educational technology research and development* 70, 2 (2022), 363–395.

[75] Ahmed Tlili, Boulus Shehata, Michael Agyemang Adarkwah, Aras Bozkurt, Daniel T Hickey, Ronghuai Huang, and Brighter Agyemang. 2023. What if the devil is my guardian angel: ChatGPT as a case study of using chatbots in education. *Smart learning environments* 10, 1 (2023), 15.

[76] Amal Trifa, Aroua Hedhili, and Wided Lejouad Chaari. 2019. Knowledge tracing with an intelligent agent, in an e-learning platform. *Education and Information Technologies* 24 (2019), 711–741.

[77] Wagino Wagino, Hasan Maksum, Wawan Purwanto, Wakhinuddin Simatupang, Remon Lapisa, and Eko Indrawan. 2024. Enhancing Learning Outcomes and Student Engagement: Integrating E-Learning Innovations into Problem-Based Higher Education. *International Journal of Interactive Mobile Technologies* 18, 10 (2024).
[78] Thiemo Wambsganss, Tobias Kueng, Matthias Soellner, and Jan Marco Leimeister. 2021. ArgueTutor: An Adaptive Dialog-Based Learning System for Argumentation Skills. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems* (Yokohama, Japan) *(CHI '21)*. Association for Computing Machinery, New York, NY, USA, Article 683, 13 pages. doi:10.1145/3411764.3445781
[79] Fei Wang, Qi Liu, Enhong Chen, Zhenya Huang, Yuying Chen, Yu Yin, Zai Huang, and Shijin Wang. 2020. Neural Cognitive Diagnosis for Intelligent Education Systems. *Proceedings of the AAAI Conference on Artificial Intelligence* 34, 04 (Apr. 2020), 6153–6161. doi:10.1609/aaai.v34i04.6080
[80] Zijie J. Wang, Chinmay Kulkarni, Lauren Wilcox, Michael Terry, and Michael Madaio. 2024. Farsight: Fostering Responsible AI Awareness During AI Application Prototyping *(CHI '24)*. Association for Computing Machinery, New York, NY, USA, Article 976, 40 pages. doi:10.1145/3613904.3642335
[81] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. 2023. Chain-of-Thought Prompting Elicits Reasoning in Large Language Models. arXiv:2201.11903 [cs.CL] https://arxiv.org/abs/2201.11903
[82] Marco Winzker. 2012. Semester structure with time slots for self-learning and project-based learning. In *Proceedings of the 2012 IEEE Global Engineering Education Conference (EDUCON)*. IEEE, 1–8.
[83] Le Wu, Xiangzhi Chen, Fei Liu, Junsong Xie, Chenao Xia, Zhengtao Tan, Mi Tian, Jinglong Li, Kun Zhang, Defu Lian, et al. 2025. EduStudio: towards a unified library for student cognitive modeling. *Frontiers of Computer Science* 19, 8 (2025), 198342.
[84] Ziwei Xu, Sanjay Jain, and Mohan Kankanhalli. 2024. Hallucination is Inevitable: An Innate Limitation of Large Language Models. arXiv:2401.11817 [cs.CL] https://arxiv.org/abs/2401.11817
[85] Andre Ye, Jared Moore, Rose Novick, and Amy X. Zhang. 2024. Language Models as Critical Thinking Tools: A Case Study of Philosophers. arXiv:2404.04516 [cs.HC] https://arxiv.org/abs/2404.04516
[86] Jifan Yu, Mengying Lu, Qingyang Zhong, Zijun Yao, Shangqing Tu, Zhengshan Liao, Xiaoya Li, Manli Li, Lei Hou, Hai-Tao Zheng, Juanzi Li, and Jie Tang. 2023. MoocRadar: A Fine-grained and Multi-aspect Knowledge Repository for Improving Cognitive Student Modeling in MOOCs. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval* (Taipei, Taiwan) *(SIGIR '23)*. Association for Computing Machinery, New York, NY, USA, 2924–2934. doi:10.1145/3539618.3591898
[87] Jifan Yu, Zheyuan Zhang, Daniel Zhang-li, Shangqing Tu, Zhanxin Hao, Rui Miao Li, Haoxuan Li, Yuanchun Wang, Hanming Li, Linlu Gong, Jie Cao, Jiayin Lin, Jinchang Zhou, Fei Qin, Haohua Wang, Jianxiao Jiang, Lijun Deng, Yisi Zhan, Chaojun Xiao, Xusheng Dai, Xuan Yan, Nianyi Lin, Nan Zhang, Ruixin Ni, Yang Dang, Lei Hou, Yu Zhang, Xu Han, Manli Li, Juanzi Li, Zhiyuan Liu, Huiqin Liu, and Maosong Sun. 2024. From MOOC to MAIC: Reshaping Online Teaching and Learning through LLM-driven Agents. arXiv:2409.03512 [cs.CY] https://arxiv.org/abs/2409.03512
[88] JD Zamfirescu-Pereira, Richmond Y Wong, Bjoern Hartmann, and Qian Yang. 2023. Why Johnny can't prompt: how non-AI experts try (and fail) to design LLM prompts. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems*. 1–21.
[89] Siyu Zha, Yuehan Qiao, Qingyu Hu, Zhongsheng Li, Jiangtao Gong, and Yingqing Xu. 2024. Designing Child-Centric AI Learning Environments: Insights from LLM-Enhanced Creative Project-Based Learning. arXiv:2403.16159 [cs.HC] https://arxiv.org/abs/2403.16159
[90] Xinlei Zhang, Takashi Miyaki, and Jun Rekimoto. 2020. WithYou: Automated Adaptive Speech Tutoring With Context-Dependent Speech Recognition. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems* (Honolulu, HI, USA) *(CHI '20)*. Association for Computing Machinery, New York, NY, USA, 1–12. doi:10.1145/3313831.3376322
[91] Chengbo Zheng, Kangyu Yuan, Bingcan Guo, Reza Hadi Mogavi, Zhenhui Peng, Shuai Ma, and Xiaojuan Ma. 2024. Charting the Future of AI in Project-Based Learning: A Co-Design Exploration with Students *(CHI '24)*. Association for Computing Machinery, New York, NY, USA, Article 94, 19 pages. doi:10.1145/3613904.3642807
[92] Yong Zheng. 2023. ChatGPT for teaching and learning: an experience from data science education. In *Proceedings of the 24th Annual Conference on Information Technology Education*. 66–72.
[93] Yihao Zhu and Qinyi Zhou. 2023. Docent: Digital Operation-Centric Elicitation of Novice-friendly Tutorials. In *Adjunct Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology* (San Francisco, CA, USA) *(UIST '23 Adjunct)*. Association for Computing Machinery, New York, NY, USA, Article 121, 3 pages. doi:10.1145/3586182.3625121
[94] Barry J Zimmerman. 2008. Investigating self-regulation and motivation: Historical background, methodological developments, and future prospects. *American educational research journal* 45, 1 (2008), 166–183.
[95] Barry J Zimmerman and Magda Campillo. 2003. Motivating self-regulated problem solvers. *The psychology of problem solving* 233262 (2003).

A Participant Recruitment Materials

A.1 Self-report Items

In the participant recruitment questionnaire, we asked the participants to report on the following items:

(1) Have you taken any courses related to analysis, and are you familiar with the calculation of derivatives, partial derivatives, and similar concepts?
(2) Have you taken any courses related to linear algebra, and are you familiar with basic matrix operations and vector operations?
(3) Can you program in Python?
(4) Have you ever run any machine learning program before?
(5) Rate your confidence in following a tutorial to complete an introductory-level machine learning project on a scale from 1 to 5 (1 for no confidence at all, 5 for complete confidence).
(6) Rate your ability to use ChatGPT on a scale from 1 to 5 (1 for never used before, 5 for highly proficient).

A.2 Validation Questions on Pre-requisites

```
Q1 - Given a function y represented by the following linear
combination:
                    y = w1·x1 + w2·x2
    Another function, L is the squared difference between y
    and a constant t:
                    L = 1/2 * (y-t)·(y-t)
    Find the partial derivative of L with respect to w1.
A) x1(y-t)
B) -x2(y-t)
C) -0.5*x1(y-t)
D) 0.5*x2(y-t)
Answer: A

Q2 - Given two vectors A=(3,-2,5) and B=(1,4,-3), find their
dot product.
A) 5
B) -20
C) -12
D) 18
Answer: B

Q3 - Point out the bug type:
    File "D:\Lab\PBL\spam_test_0823\main.py", line 61, in
    synonym_replacement word to replace =
    random.choice(random word list)
    File "D: \Miniconda\envs\pbl-test\Lib\random.py", line
    373, in choice, raise IndexError('Cannot choose from an
    empty sequence' ) IndexError:Cannot choose from an empty
    sequence
A) Attempting to read a null reference.
B) An index error occurred because random_word_list is an
empty list.
C) Attempting to access word_to_replace before it is defined.
D) The random.choice() function does not accept a list as
input.
Answer: B
```

A.3 Testing Questions on Prior Knowledge of Machine Learning

```
Q1 - Which of the following methods are commonly used to
mitigate overfitting?
A) Reducing the number of model parameters
B) Increasing model depth
```

C) Using regularization
D) Reducing the amount of training data
E) Self-supervised learning
Answer: A and C

Q2 - Please select all the correct options from the following descriptions about machine learning:
A) Support Vector Machines (SVM) can effectively classify non-linearly separable data using a kernel function.
B) In decision trees, the closer a feature is to the root node, the more information it contains for classification and the higher its discriminative power.
C) Support Vector Machines (SVM) perform poorly in high-dimensional data and are, therefore, only suitable for classification in low-dimensional data.
D) Principal Component Analysis (PCA) can reduce the dimensionality of input data, thereby reducing computational power consumption.
E) One of the reasons for the success of deep learning is that neural networks can learn features automatically.
Answer: A, D, and E

Q3 - In deep learning, the learning rate is an important hyperparameter. Please select all the correct statements regarding the learning rate from the following options:

A) The learning rate refers to the proportion of samples involved in gradient computation during a single backpropagation.
B) A higher learning rate may cause difficulties in the convergence of the training process.
C) A lower learning rate makes it less likely for the model to converge to a local optimum.
D) Modern optimization algorithms like Adam and AdamW can adjust the learning rate adaptively to some extent.
E) Although the optimal learning rate needs to be found through multiple trials, in general, setting the learning rate to 1e-4 is a good starting point.
Answer: B, D, and E

Q4 - Skip connections (residual connections) are widely used in various neural network architectures. Please select all the correct statements about skip connections from the following:
A) Skip connections aim to address performance issues in deep neural networks.
B) In this context, "residual" refers to the result of subtracting the output value of a layer from its input value.
C) The theoretical basis of skip connections is that neural networks struggle to learn the identity transformation during training.
D) Skip connections are widely used in recurrent neural networks (such as RNNs and LSTMs).
E) Skip connections are an important part of the Transformer architecture.
Answer: A, C, and D

B Quiz Questions

We present the questions used in the closed-book quizzes to assess learning outcomes here.

B.1 Spam Classification

Q1 - In spam detection, which text vectorization technique should we use to consider the frequency of a word in the corpus?
A) Word Embedding
B) Bag of Words
C) TF-IDF
D) One-Hot Encoding
Answer: C

Q2 - Which option best describes how SVMs work?
A) Maximizing the number of support vectors.
B) Finding a hyperplane with maximum margin.
C) Finding a support vector that minimizes its distance from other samples.
D) Minimizing the distance between support vectors and other samples.
Answer: B

Q3 - Suppose you have a dataset containing 1000 samples and 5000 features. You split the dataset 80/20 into a training set and a test set. Which of the following descriptions about the shape of the datasets after the split is correct?
A) Train set: (1000, 4000), test set: (1000, 1000)
B) Train set: (800, 5000), test set: (200, 5000)
C) Train set: (800, 1000), test set: (200, 1000)
D) Train set: (800, 4000), test set: (200, 1000)
Answer: B

Q4 - When using a Support Vector Machine (SVM) for text classification, which of the following descriptions about linear kernels is correct?
A) Linear kernels are suitable for data that cannot be linearly separated in high-dimensional space.
B) Linear kernels have relatively low computational efficiency and are suitable for handling complex non-linear relationships.
C) Linear kernels are suitable for data in high-dimensional feature spaces, especially when the number of features far exceeds the number of samples.
D) Linear kernels perform poorly in text classification problems and are generally less effective than polynomial kernels.
Answer: C

Q5 - John is applying for PhD and does not want to have any message related to the application process sent to the spam box. Which metric of the spam detection algorithm concerns John the most?
(We define spam messages as positive samples in this question)
A) Accuracy
B) Recall
C) Precision
D) F1 Score
Answer: C

Q6 - Which line of code can generate predictions using a trained model?
A) predictions = model.predict(X_test)
B) predictions = model.transform(X_test)
C) predictions = model.fit(X_test)
D) predictions = model.score(X_test)
Answer: A

Q7 - Which of the following statements about grid search is correct?
A) Grid search optimizes the model by randomly choosing hyperparameter combinations.
B) Grid search is capable of optimizing several hyperparameters simultaneously.
C) Grid search is suitable only for optimizing linear models.
D) Grid search can only be used to optimize the regularization parameter C.
Answer: B

Q8 - Which of the following factors should be considered when choosing the kernel function of SVMs?
A) Whether the dataset can be linearly divided
B) The size of the dataset
C) The number of support vectors needed
D) The computing resources needed
Answer: A, B, and D

Q9 - Which of the following feature engineering practices might work in spam detection?
A) TF-IDF

```
B) Choosing the most frequent word as the main feature
C) Using the existence of certain words or phrases as a
feature
D) Using the length of the message as a feature
Answer: A, C and D

Q10 - Which of the following lines of code can be used to
pre-process data for spam detection?
A) data['message'] = data['message'].dropna()
B) data['message'] = data['message'].fillna(-1)
C) X = TfidfVectorizer().fit_transform(X)
D) X = MinMaxScaler().fit_transform(X)
Answer: A, C
```

B.2 Digit Recognition

```
Q1 - What is the primary function of the convolutional layer
in a Convolutional Neural Network (CNN)?
A) Extracting features
B) Reducing feature dimensions
C) Reducing overfitting
D) Introducing non-linearity
Answer: A

Q2 - Why are the images in the MNIST dataset 28x28 pixels?
A) 28x28 is the optimal kernel size.
B) Smaller images reduce the computational load.
C) Most handwritten digits are easier to recognize in 28x28
images.
D) 28x28 is the optimal input shape of CNNs.
Answer: B

Q3 - What does data augmentation do?
A) Increasing the amount of training data.
B) Speeding up the training process.
C) Increasing the number of parameters in the model.
D) Making the model converge faster during training.
Answer: A

Q4 - Which line of code normalizes image data?
A) x = x.view(-1, 64*7*&)
B) x = F.relu(self.conv1(x))
C) torch.nn.functional.normalize(x, p=2.0)
D) x = x / 255.0
Answer: C

Q5 - Which of the following statements about cross-entropy
loss is correct?
A) Cross entropy loss is unsuitable for binary
classification.
B) Cross entropy loss has been implemented in PyTorch as
torch.nn.MSELoss.
C) Softmax cannot be applied before calculating
cross-entropy loss.
D) Cross entropy loss is used in image classification
instead of MSE loss because it better distinguishes the
difference between classes.
Answer: D

Q6 - If a model trains very slowly, with very small (almost
zero) gradient updates in each batch, what is the most
probable cause?
A) Unsuitable activation functions.
B) Improper regularization techniques.
C) The learning rate is too low.
D) The learning rate is too high.
Answer: A

Q7 - Which of the following terms best describes the case
where a model performs poorly on both the train and test
sets?
A) Underfitting
B) Generalizable
C) Overfitting
D) Regularization
Answer: A

Q8 - Which of the following components make up CNNs?
A) Fully connected (or linear) layers
B) Convolutional layers
C) Recurrent layers
D) Pooling Layers
Answer: A, B, and D

Q9 - Which of the following metrics is suitable for the
MNIST dataset?
A) Accuracy
B) Area under curve (AUC)
C) Mean absolute error
D) Confusion matrix
Answer: A and D

Q10 - Which lines of code update model parameters?
A) optimizer.step()
B) loss.backward()
C) model.zero_grad()
D) model.fit(X_train, y_train)
Answer: A and D
```

C Questionnaire

We presented the participants with a questionnaire to collect their quantitative subjective feedback. All the questionnaire questions were asked on a 7-point scale.

C.1 Perceived Gains

1 for *I have learned nothing*, 7 for *I have learned a lot.*

(1) Conceptual Understanding
(2) Practical Skills
(3) Understanding of ML as a whole

C.2 NASA TLX

1 for *very low*, 7 for *very high*

(1) How mentally demanding was the task?
(2) How physically demanding was the task?
(3) How hurried or rushed did you feel during the task?
(4) How successful do you think you were in accomplishing the task?
(5) How hard did you have to work to complete the task?
(6) How insecure, discouraged, irritated, or stressed did you feel during the task?

C.3 Metacognition on Learning ML with LLMs

1 for *strongly disagree*, 7 for *strongly agree*

(1) I asked good questions for learning.
(2) I made many deep inquiries
(3) I collaborated efficiently with AI
(4) I am confident learning ML with LLM
(5) I am confident doing self project-based learning with LLM

C.4 Usability

1 for *strongly disagree*, 7 for *strongly agree*

(1) I am satisfied with the general experience.
(2) I want to use it in the future.
(3) It is easy to use.
(4) It was a pleasant experience.

C.5 Feature Usability (for AutoPBL)

We asked participants how they liked the following key features (1 for no use at all, 7 for highly useful): (1) Sidebar Table of Contents, (2) Bite-sized Blocks, (3) Adaptive Content, (4) Checkpoint Questions, (5) 20-second Countdown, (6) Virtual TA, (7) Quote & Ask, (8) Preset Inquiry Modes.

D Inquiry Behavior Analysis Codebook

We present the codebooks for inquiry behavior analysis. Table 2 is the codebook for inquiry content, while Table 3 is for the motivation behind inquiries.

E Interview Script

In the post-session interviews, we asked participants the following questions:

(1) Can you evaluate your recent learning experience? What knowledge do you feel you have gained? What skills have you acquired? Have there been any changes in your understanding or perception?
(2) Please evaluate the SPBL process. How did you carry out your PBL learning? What was your internal state during the process? What did you do with the LLM chatbot?
(3) Do you think the recent method was effective? How did you feel while using it? Which features helped you the most in completing the project and learning? Which features confused you?
(4) How does the experience of using AutoPBL compare to the combination of tutorials + ChatGPT? What aspects were better? What aspects were worse?

F Prompts

F.1 Project Designer

Agent Functionality: Generate the framework (steps and sub-steps) of a project based on the learner's needs and profile.

System Prompt:

```
{
  "role": "You are an expert in project-based learning. You
  specialize in teaching AI and deep learning through
  projects."
  "task": "Design the framework of an AI/DL tutorial
  according to the needs and interests of the learner. Take
  the learner's level of knowledge into consideration in
  your design."
  "requirements": [
    "The project must be self-contained, covering the entire
    process from data processing to testing and evaluation.",
    "The framework should be divided into steps, which in
    turn is divided into sub-steps. For example, the step
    Data Processing might contain the sub-steps
    Visualization, Data Processing, and Data Augmentation.",
    "The tutorial should not be about finishing a project
    only. It must be designed to teach knowledge, concepts,
    common practices, and tricks about the relevant field of
    AI and deep learning.",
    "Step 0 must be an introduction to the project and what's
    being taught, its background and history, and its
    real-world applications. Also, explain in layman's words
    what the learner will do in the tutorial.",
    "Beginning from step 1, sub-step 0 of each step must be
    about the background and historical information in
    layman's terms. Interesting anecdotes are welcomed, too.
    The title must be \"Background Information\".",
    "Beginning from step 1, sub-step 1 from each step must be
    related knowledge, theories, and concepts related to
    this step. The title must be \"Concepts\". For example,
    if the step is about defining a neural network model, go
    through the basic concepts like artificial neurons, MLP,
    gradients, and backpropagation.",
    "You can break down each step into many (ideally 3 to 6)
    sub-steps. The division can be more fine-grained. The
    content of each step should be detailed and well
    organized.",
    "For deep learning projects, use PyTorch. For
    traditional ML projects, use Scikit-Learn. The learner
    is using Visual Studio Code as the IDE.",
    "Whenever possible, use common, readily available
    datasets in the tutorial. You must clearly indicate
    which dataset to use in the overview section of the data
    processing step.",
    "You only need to give an overview of the steps involved
    in the project. You do not need to generate detailed
    contents or code for each step.",
    "You must use the following JSON format: {\"title\":
    (the title of the tutorial), \"step_cnt\": (the number
    of steps), \"steps\": [{\"id\": (index, starts from 0),
    \"name\": (name of the step), \"overview\": (a brief
    overview of this step), \"sub_step_cnt\": (number of
    sub-steps), \"sub_steps\": [{\"id\": (index, starts from
    0), \"name\": (name of sub-step), \"overview\": (a brief
    overview of this sub-step)}, ...]}, ...]}"
  ]
}
}
```

F.2 Block Content Generator

Agent Functionality: Respond to the learner's answer to the previous checkpoint question and generate the content of the next block.

System Prompt:

```
{
  "role": "You are an expert in project-based learning. You
  specialize in teaching AI and deep learning through
  projects.",
  "task": "Given the overall framework of a tutorial and the
  learner's current progress, generate the next part of the
  tutorial. The purpose of the tutorial is to teach the
  learner, not just to speed through the project, so you
  should include the key concepts, maths, theories, and
  other details in your content when appropriate."
  "requirements": [
    "We divide the tutorial into steps, steps into sub-steps,
    and sub-steps into blocks. Blocks are bite-size pieces
    that are atomic in the sense of user interaction,
    meaning that the learner only needs to respond at the
    end of each block. You only need to generate the next
    block in the tutorial. Your block must correspond to one
    sub-step of a step only and must not cover any other
    sub-steps of the framework.",
    "Compared to the previous block, you may stay in the same
    sub-step or move into the next sub-step. You must not
    skip any sub-steps. For example, you may move from step
    0 sub-step 0 into step 0 sub-step 1.",
    "There are two types of blocks: tutorial (contents of the
    tutorial, including text and code) and gpt_follow_up
    (follow-up to the user's answer to your questions).",
    "You must generate the appropriate type of block
    according to the framework of the tutorial, the current
    context, and previous blocks.",
    "DO NOT TEACH OR CODE DURING STEP 0. THE INTRODUCTION
    STEP IS ONLY FOR PROVIDING BACKGROUND AND INVOKING
    INTEREST IN LAYMAN'S TERMS.",
    "YOU MUST FOLLOW THE FRAMEWORK OF THE TUTORIAL. YOU MUST
    NOT DEVIATE FROM IT.",
    "You must ensure continuity with the previous block.",
```

Category	Code	Example
Understanding	Asking for an explanation of concepts or math	What is a *hyperplane*?
Execution	Asking for code explanation	What does the argument p means in RandomHorizontalFlip(p=0.5)?
Understanding	Asking directly for checkpoint question solutions (conceptual)	Answer this question: Why does maximizing the margin improve generalization when using SVMs for classification tasks?
Execution	Asking directly for checkpoint question solutions (coding)	Finish the TODO sections of the code I quoted.
Execution	Asking about Python package usage	How can I use the forward function in PyTorch?
Execution	Requesting code as specified by the user	Provide an example of applying weight initialization to a model.
Execution	Asking for debug support	I got this error message. What's wrong? ValueError: Found input variables with inconsistent numbers of samples: [1115, 4457]
Understanding	Asking for visualization	Visualizing a decision boundary for me.
Understanding	Asking for confirmation of statement	Is it OK to assume that most samples do not influence the decision boundary in SVMs?
Understanding	Asking exploratory questions	Can we go deeper into the math behind the Adam optimizer?
Irrelevant	Irrelevant question	Hi, how are you doing?

Table 2: Inquiry Content Codebook

Code	Example
Debug-Driven	I got this error message. What's wrong? ValueError: Found input variables with inconsistent numbers of samples: [1115, 4457]
Self-Driven	What does the argument p mean in RandomHorizontalFlip(p=0.5)?
Checkpoint-Driven	Answer this question: Why does maximizing the margin improve generalization when using SVMs for classification tasks?
Irrelevant	Hi, how are you doing?

Table 3: Inquiry Motivation Codebook

```
"You should go a bit deeper into how the models and
algorithms work. For example, you should teach through
math and simple examples how neurons, linear layers,
convolution kernels, backpropagation, and word
embeddings work. Get into the specifics, such as how to
calculate convolution in CNNs, how to do TF-IDF, how to
calculate gradients, etc."
"Each block must have a defined purpose, and the purpose
must be chosen from one of the following - invoking
learner's interest, teaching knowledge and concepts,
explaining the step or sub-step from the top down,
setting up environment or dataset, providing the coding
for the next task, providing directions for further
exploration (such as more model architectures or tricks),
invoking reflection on what has been learned and done,
and providing code for visualizing concepts (such as
positional encoding, data augmentation, feature
engineering, etc.),"
"Each block must have one and only one purpose. You MUST
NOT mingle together for 2 different purposes. For
example, a block must not simultaneously introduce new
concepts and provide code."
"When invoking interest, provide additional background
information or even intriguing anecdotes to keep the
learner's attention.",
"When providing the coding for the next task, if the code
is related to the core part of the model or algorithm
taught in the tutorial, leave the most important parts
blank and let the learner figure out how to do it
themselves. These blanks should be marked by a comment
line in this format: #TODO: Define a Conv2d layer. When
generating the code snippet, leave out the lines of code
corresponding to the TODO comment. You may explain how
to write the code for these parts but must not give the
code to the user directly. You may provide complete code
for dirty work like data preprocessing."
"When teaching knowledge and concepts, you can give a
general direction and have the learner try to come up
with the rest. For example, you can give them an example
of how to calculate the gradient for the output layer
and let the learner try to figure out how to calculate
the gradient of the hidden layer.",
"Do not end the contents in a question. The interactive
section of the block will be dealt with later.",
"Invoke reflection at the end of each step, except for
the introduction step and environment setup step.",
```

```
    "You must use the following JSON format: {\"step\": (the
    step this block belongs to), \"sub_step\": (the sub-step
    this block belongs to), \"block_type\": (tutorial,
    gpt_follow_up or gpt_answer), \"purpose\": (the purpose
    of the block, chosen ONLY from the ones listed above),
    \"needs_coding_by_learner\": (1 if this block needs the
    learner to independently fill #TODO lines, 0 if this
    block provides code in entirety or is not related to
    code at all), \"learner_coding_task\": (If
    needs_coding_by_learner is 1, explain which core parts
    need to be coded independently by the learner and should
    be marked as #TODO in the code given.), \"content\":
    (the content of this block in text)}",
    "All math formulas should be written in KaTeX format and
    surrounded with dollar signs ($ or $$)."
    "All hyperlinks should be written in markdown format
    like this: [link text](link URL)."
    "Include only one block in your response! Write one JSON
    object and NOTHING ELSE. DO NOT WRITE ANYTHING OTHER
    THAN THE JSON OBJECT! USE DOUBLE QUOTES (\") IN JSON!",
    "You only have two options with the step and substep.
    After considering the context, you can either 1) STAY
    WITHIN THE SAME SUBSTEP AS THE PREVIOUS BLOCK or 2) MOVE
    ON TO THE NEXT SUBSTEP AS GIVEN IN THE TUTORIAL
    FRAMEWORK. You MUST NOT move to a previous substep or
    skip substeps altogether."
  ]
}
```

F.3 Checkpoint Question Generator

Agent Functionality: Generate a checkpoint question based on the contents of a block.

System Prompt:

```
{
  "role": "You are an expert in project-based learning. You
  specialize in teaching AI and deep learning through
  projects.",
  "task": "Given part of a tutorial, as well as the overall
  framework of the entire tutorial, generate an appropriate
  interactive section for the user to improve the outcome of
  the tutorial.",
  "requirements": [
    "We divide the tutorial into steps, steps into sub-steps,
    and sub-steps into blocks. Blocks are bite-size pieces
    that are atomic in the sense of user interaction,
    meaning that the learner only needs to respond at the
    end of each block. You only need to generate the next
    block in the tutorial. Your interactive section must
    correspond to the block given to you.",
    "There are two types of blocks: tutorial (contents of the
    tutorial, including text and code) and gpt_follow_up
    (follow-up to the user's answer to your questions).",
    "Each block has a defined purpose, chosen from one of the
    following - invoking learner's interest, teaching
    knowledge and concepts, explaining the step or sub-step
    from the top down, setting up the environment, providing
    the coding for the next task, providing directions for
    further exploration (such as more model architectures or
    tricks), invoking reflection on what has been learned
    and done, and providing code for visualizing concepts
    (such as positional encoding, data augmentation, feature
    engineering, etc.),"
    "The interactive section must enhance the purpose and
    function of the block given to you. When invoking
    interests, you may ask the user to envision potential
    applications or ask if they want to know more about how
    the concept or technique taught can be used in practical
    applications.",
    "When teaching knowledge of concept, you may ask the user
    to try and design their solutions and compare them to
    the standard answer, or ask them questions about
    theories and tricks.",
    "When setting up environment, you want to make sure the
    learner got the environment right.",
    "When teaching the top-down overview of some pipeline or
    practice, ask the learner to reiterate the rationale
    behind the top-down design.",
    "When presenting code, ask the learner to fill in the
    blank parts and give back the complete code, ask them to
    provide the results from running the code, or ask them
    to explain core parts of the code given.",
    "When directing further exploration, you may want the
    user to report what they observed during the exploration
    and ask them to explain the observed results or ask them
    to provide their intended direction for the next step of
    exploration.",
    "When invoking reflection, ask the learner to reflect how
    and why they did what they did.",
    "When providing visualization, ask them what they have
    seen or whether they gained a clearer understanding of
    the concept being visualized.",
    "YOU MUST FOLLOW THE FRAMEWORK OF THE TUTORIAL. YOU MUST
    NOT DEVIATE FROM IT.",
    "You don't need to ask for detailed questions every time.
    Sometimes a single_choice question asking for
    acknowledgment that the learner is following the
    tutorial is enough.",
    "Please do not ask questions that are too obvious or too
    easy. You should guide the learner to think actively
    when possible.",
    "To provoke active thinking, do not ask the learner to
    repeat or reiterate what has been taught simply. Guide
    them to go deeper and think harder.",

    "Do not insult the learner by asking questions with an
    obvious answer already present in the contents of the
    block. If you have nothing meaningful to ask, use the
    single_choice option and ask for a single acknowledgment.
    For example, if the block tells the user basic facts
    like what they will do in the project or what a certain
    dataset contains, you only need to ask a single_choice
    question for acknowledgment. In this case, the single
    choice being something like \" Understood\" is enough.",
    "Apart from factual questions, you may also ask about
    whether the learner wants to explore some more details.",
    "There are three types of user interaction section:
    multi_choice (where the user may choose ONE of several
    options given by you), single_choice (where the user can
    acknowledge a statement given by you), and text_input
    (where the user can respond by text).",
    "Your output must be in one of the following JSON
    formats.",
    "\"multi_choice\" should follow this JSON format:
    {\"interative_section_design_rationale\": (the rationale
    behind the design of the interactive section), \"desc\":
    (description of the multi-choice question for the user.
    The question must have only one correct choice among the
    several choices), \"type\": \"multi_choice\,"
    \"choices\": [{\"choice_index\": (the index of choice,
    starting from 0), \"choice_content\": (the content of
    this choice)}, ...]}",
    "\"single_choice\" should follow this JSON format:
    {\"interative_section_design_rationale\": (the rationale
    behind the design of the interactive section), \"desc\":
    (description of the single choice question for the user),
    \"type\": \"single_choice\", \"choice\": (the content of
    choice)}",
    "\"text_input\" should follow this JSON format:
    {\"interative_section_design_rationale\": (the rationale
    behind the design of the interactive section), \"desc\":
    (description of the question for the user), \"type\":
    \"text_input\"}",
    "All math formulas should be written in LaTex format and
    surrounded with dollar signs ($ or $$).,
    "All hyperlinks should be written in markdown format
    like this: [link text](link URL)."
    "Include only one interactive section in your response!
    Write one JSON object and NOTHING ELSE. DO NOT WRITE
    ANYTHING OTHER THAN THE JSON OBJECT! USE DOUBLE QUOTES
    (\") IN JSON!"
```

```
  ]
}
```

F.4 Summarizer

Agent Functionality: Summarize the current progress based on the project framework and blocks.

System Prompt:

```
{
  "role": "You are an expert in project-based learning. You
  specialize in teaching AI and deep learning through
  projects.",
  "task": "Summarize what has happened in the tutorial so
  far. The framework is for reference only. If a step in the
  framework is not in the context given to you, you should
  not put it in the summary.",
  "requirements": [
    "We divide the tutorial into steps, steps into sub-steps,
    and sub-steps into blocks.",
    "There are four types of blocks: tutorial (contents of
    the tutorial, including text and code), gpt_follow_up
    (follow-up to the user's answer to questions),
    user_question (the learner's questions about tutorial
    contents), and gpt_answer (the answer to the learner's
    questions).",
    "Summarize the blocks given to you, focusing on what each
    step (denoted in the block by step_index) taught, what
    the learner did in that step, and what question, if any,
    the learner raised during the step.",
    "Be brief and concise. The output should be a list of
    summaries, one for each step.",
    "The framework of the tutorial may contain steps yet to
    be done. Ignore these and focus on the contents of the
    blocks given to you."
  ]
}
```

F.5 Code Masking Agent

Agent Functionality: Remove excess code from the content of a block. This prevents AutoPBL from giving away the answers to code completion tasks.

System Prompt:

```
{
  "role": "You are an expert in project-based learning. You
  specialize in teaching AI and deep learning through
  projects.",
  "task": "Given part of a tutorial, remove the lines marked
  with #TODO and leave only the #TODO comment behind,
  creating a code completion task for the learner.",
  "requirements": [
    "We divide the tutorial into steps, steps into sub-steps,
    and sub-steps into blocks. Blocks are bite-size pieces
    that are atomic in the sense of user interaction,
    meaning that the learner only needs to respond at the
    end of each block.",
    "You are given a block described in JSON. Remove lines
    marked with #TODO comments and keep everything else the
    same. You MUST keep the JSON format the same and
    everything else apart from the \"content\" field the
    same.",
    "You can modify the #TODO comments to explain in greater
    detail what needs to be done and even link to
    documentation websites for things like NumPy, sklearn,
    and PyTorch.",
    "Do not just comment out the lines that do the function
    entailed in #TODO comments. Delete them entirely. You
    must not show the learner the standard solution when
    they are supposed to independently finish these parts of
    the code.",
    "If there are no #TODO comments in the block given to
    you, keep the output the same as the JSON given to you."
  ]
}
```

F.6 Virtual TA

Agent Functionality: A conversational virtual TA assisting the learner.

System Prompt:

```
{
  "role": "You are an expert in project-based learning. You
  specialize in teaching AI and deep learning through
  projects.",
  "task": "The learner wants to discuss some content in the
  tutorial with you. You will be given the framework of the
  tutorial, a summary of the learner's learner's current
  progress, and the content they have questions about."
  "requirements": [
    "Be engaging, helpful, and ready to answer questions as
    long as they relate to the tutorial. Do not give away
    the full answer to a complex question right away. Guide
    the learner to think first. Progressively provide more
    assistance if the learner has trouble figuring out the
    problem on their own.",
    "If the learner deviates too much from the tutorial,
    remind them to stay on track.",
    "Encourage the learner when needed, such as when they
    have trouble fixing a bug.",
    "All math formulas should be written in LaTex format and
    surrounded by dollar signs ($ or $$).",
    "All hyperlinks should be written in markdown format
    like this: [link text](link URL)."
  ]
}
```