IRSTI 28.23.37
DOI: https://doi.org/10.47344/88hpey93

Gaukhar Duisenova[1], Nurzhan Shyndaliyev[2], Rustam Shadiev[3]*
[1,2] L.N. Gumilyov Eurasian National University, Astana, Kazakhstan
[3] Zhejiang University, Hangzhou, China
*e-mail: gauhar.duisen9@gmail.com

AI-PERSONALIZED PROGRAMMING TASKS BASED ON DIGITAL TRACE ANALYSIS: A PILOT STUDY WITH PRE-SERVICE COMPUTER SCIENCE TEACHERS IN A CONTAINERIZED LAB ENVIRONMENT

Abstract. Pre-service computer science teacher education must simultaneously develop programming competence and pedagogical content knowledge, yet no empirical study has evaluated whether AI-driven task personalization based on digital trace analysis can support this dual objective in a containerized laboratory environment. This paper presents a mixed method quasi-experimental pilot study involving 32 pre-service informatics teachers in two university cohorts in Kazakhstan. Participants in the experimental group (n = 17) received AI-personalised programming tasks generated from a real-time analysis of their digital footprints in Docker-based virtual laboratories; control groups (n = 15) completed the same tasks in the same environment without customization. Over eight weeks, the experimental group achieved statistically significantly higher gains in both programming skills assessment (d = 0.71, p = 0.012) and validated TPACK survey instruments (d = 0.58, p = 0.031), with particularly strong effects in teaching content knowledge subscales (d = 0.84). Qualitative analysis of post-study interviews revealed that participants valued the transparency of task recommendations but expressed concern about algorithmic fairness and the accuracy of the system's assessment of their pedagogical knowledge. These findings provide the first preliminary empirical evidence that TPACK-aligned AI personalization from digital traces is feasible and effective for pre-service informatics teacher training, while identifying concrete design improvements for subsequent iterations.

Keywords: AI personalization, digital trace analysis, pre-service teachers, programming education, containerized virtual laboratory, TPACK, mixed-methods pilot study, learning analytics, Kazakhstan

Introduction

The dual challenge facing pre-service informatics teacher education is widely acknowledged but poorly served by current instructional technology. Existing intelligent tutoring systems and adaptive programming platforms optimize exclusively for skill acquisition in general computing student populations. They offer no scaffolding for the pedagogical knowledge development that distinguishes teacher preparation from ordinary computing education (Saeli et al., 2011; Mouza et al., 2021). At the same time, the behavioral data produced by modern containerized programming laboratories contains rich evidence of both knowledge types. Compilation events and keystroke patterns reflect programming content knowledge, while code annotations and help-seeking strategies reflect pedagogical self-awareness. No deployed system exploits this multi-signal stream for TPACK-aligned personalization (Ihantola et al., 2015; Azcona et al., 2019).

This gap has real implications in the educational setting of Central Asia where this research is focused. The national informatics curriculum of Kazakhstan was significantly extended after the 2019 Digital Kazakhstan state program, which led to a severe shortage of qualified informatics teachers. The teachers' training has to be carried out efficiently within the time limits of the university programs (Ministry of Digital Development, 2017). The pre-service teacher candidates in Kazakhstani universities have very diverse programming backgrounds when they come into the informatics teacher programs. Some have previous competition programming experience while others have the

minimum exposure at best. Therefore adaptive personalized instruction is not simply a luxury that one can do without but is structurally an absolute necessity for equitable preparation outcomes even within a single cohort (Yadav et al., 2017).

The present study evaluates a prototype implementation of an adaptive educational system (AES) architecture, grounded in the four-component model of Brusilovsky and Peylo (Brusilovsky & Peylo, 2003), extended to incorporate TPACK-aligned learner modeling for pre-service teacher contexts. The study addresses three research questions. RQ1 asks whether AI-personalized task delivery based on digital trace analysis produces significantly greater programming skill gains than standard instruction in the same containerized environment. RQ2 asks whether AI personalization based on TPACK-aligned trace analysis produces significantly greater TPACK development than standard instruction, particularly on the Pedagogical Content Knowledge subscale. RQ3 asks how pre-service informatics teacher candidates perceive AI-driven task personalization and what design implications emerge from their perceptions.

The paper is organized as follows. Section 2 reviews the empirical literature directly relevant to the study design. Section 3 describes the prototype system, study participants, and methodology in detail. Section 4 presents quantitative and qualitative results. Section 5 discusses findings in the context of the broader literature and identifies design implications. Section 6 concludes.

Literature Review

AI Personalization in Programming Education: Evidence Base

The strongest empirical evidence for AI-driven personalization in programming education comes from data-driven hint generators and knowledge-tracing-based adaptive sequencing systems. Rivers and Koedinger's ITAP system demonstrated in a randomized study that data-driven next-step hints generated from aggregated solution paths improved task completion rates and reduced subsequent error rates compared to no-hint conditions, with particularly strong effects for students in the middle tercile of prior knowledge (Rivers & Koedinger, 2017). Aleven et al. carried out research over time on the behaviour of raising a hand for help in intelligent tutoring systems. They found that the effectiveness of computer-generated help is largely determined by the level of control the learner has over the situation. Giving students hints without them asking for it makes them perform better temporarily but discourages them from finding solutions on their own in the future, whereas hints that are asked for at the right moment help to develop both these skills (Aleven et al., 2016). The results of this research are the basis for one of our changes we made in this research, i.e. showing AI-generated recommendations as optional with a clear explanation of the reasoning behind them rather than forced task assignments.

Since 2023, LLM-based methods to programming task generation and personalization have rapidly increased in number. Denny and his colleagues showed that GPT-4-style models are capable of generating programming exercises that are syntactically correct and pedagogically aligned with different levels of difficulty and topic coverage. However, they observed that the models tend to create exercises that target general skill areas rather than specific student misunderstandings (Denny et al., 2024). Prather et al.'s comprehensive working group report identified over-reliance on AI-generated content as a significant risk in programming education. They recommended that AI task generation be integrated within a human-supervised curriculum framework, a recommendation incorporated in the present study's design (Prather et al., 2023).

Digital Trace Analysis for Learner Profiling

Digital trace-based learner profiling in programming education has reached sufficient maturity to support reliable risk prediction and strategy characterization in general computing student populations. Azcona, Hsiao, and Smeaton's longitudinal study of 211 students demonstrated that digital footprint features like submission timing, compilation frequency, and error recovery latency predicted semester outcomes with AUC = 0.84 from week three onward (Azcona et al., 2019). Zhang, Shi, and Huang more recently showed that keystroke-derived features explained 31% of variance in expert-rated code quality scores independently of compilation correctness. They also established that trace data carries information about programming process quality beyond simple correctness metrics

(Zhang et al., 2023). Neither study involved teacher candidates, and neither connected trace features to TPACK dimensions.

TPACK Development in Pre-Service Informatics Teachers

Empirical studies of TPACK development in pre-service informatics teacher programs consistently find that Pedagogical Content Knowledge is the component that develops most slowly and requires the most deliberate instructional scaffolding (Saeli et al., 2011; Mouza et al., 2021). Villegas-Ch, Garcia-Ortiz, and Sanchez-Viteri did a systematic review to see how AI and learning analytics were used in teacher education. They only found three studies that tried to use behavioral data to help develop TPACK, and none of those studies were about informatics teachers or programming contexts (Villegas-Ch et al., 2022). Standl and Brinda conducted a design study of teaching-learning laboratories in informatics teacher education. They recognized reflective task design and peer discussion of learner misconceptions as the most effective pedagogical strategies for PCK development. However, they carried out these strategies by manual instructor intervention rather than automated systems (Standl & Brinda, 2020).

Adaptive Educational System Architectures

Adaptive educational systems have been formally theorized for nearly three decades. Brusilovsky and Peylo's four-component AES architecture organizes adaptive instruction around a domain model, a learner model, an adaptation model, and a presentation layer (Brusilovsky & Peylo, 2003). The first model is about structured knowledge to be taught, second one is continuously updated estimates of the learner's knowledge state, third rules mapping learner states to instructional actions, and finally the interface through which adapted content is delivered. This architecture has been validated across multiple educational domains and remains the dominant design template for intelligent adaptive systems in formal education. Its application to teacher education contexts, where the domain model must encode both content knowledge and pedagogical knowledge simultaneously, has not been systematically explored. The present study instantiates this architecture within a containerized programming laboratory environment and extends the learner model to track TPACK component mastery independently.

Method

Research Design

The study employed a mixed-methods quasi-experimental design with pre-test and post-test measurements and a concurrent qualitative strand. Because of the institutional setup of the participating programs, a quasi-experimental design rather than a fully randomized one was required: participants were divided into experimental and control groups according to the information of their enrollment in one of two parallel sections of the same course at two different partner universities. University A was assigned to the experimental condition and University B to the control condition prior to participant enrollment, based on differences in local infrastructure readiness for container orchestration deployment rather than on any participant-level characteristic. This was done to reduce the selection bias while taking into account the institutional limitations. Two distinct institutional settings, and unmeasured differences in instructor style, campus culture, or program emphasis may have contributed to observed outcome differences independently of the intervention. A qualitative strand of the research was a set of post-study semi-structured interviews with a purposive subsample of 12 participants, where 6 experimental and 6 control were selected to depict the entire spectrum of programming skills and gender identities present in the sample.

Participants

Thirty-two pre-service informatics teacher candidates participated in the study across two universities in Kazakhstan. University A contributed 17 participants to the experimental group (10 female, 7 male; mean age 20.3 years, SD = 1.2); University B contributed 15 participants to the control group (9 female, 6 male; mean age 20.7 years, SD = 1.4). All participants were enrolled in their second year of a four-year Bachelor of Education (Informatics) program and were taking their first structured Python programming course as part of the curriculum. Pre-test scores on the programming skills assessment ($M_{exp} = 41.3$, $SD = 8.7$; $M_{ctrl} = 42.1$, $SD = 9.2$) and TPACK

survey (M_exp = 3.21, SD = 0.44; M_ctrl = 3.18, SD = 0.51) did not differ significantly between groups (programming: t(30) = 0.27, p = 0.79; TPACK: t(30) = 0.18, p = 0.86). It confirms group equivalence at baseline. Ethical approval was obtained from the institutional review boards of both universities. All participants provided written informed consent prior to enrollment.

The Adaptive Programming Laboratory System

The prototype system instantiates Brusilovsky and Peylo's four-component AES architecture (Brusilovsky & Peylo, 2003) within a Kubernetes-orchestrated deployment, allocating one Docker container per participant. Every container hosted a JupyterLab instance as the main programming interface, equipped with a specially-designed Python telemetry plugin that recorded keystrokes, cell execution attempts, and output streams. The container runtime was set to Docker Event API logging system to track filesystem operations and process lifecycle events at the event capture level. A shared ELK (Elasticsearch-Logstash-Kibana) pipeline collected events from all the containers in real-time, using TPACK-aligned feature extraction rules to identify the TPACK component each behavioral signal most directly represents.

The learner model part kept four separate Bayesian Knowledge Tracing (BKT) chains per participant (Corbett & Anderson, 1994), with one for each of the main TPACK components targeted in the course: Content Knowledge, Technological Knowledge, Pedagogical Knowledge, and Pedagogical Content Knowledge. After each work session, these chains were updated following the usual BKT equations. The personalization decision followed a fixed sequence. The adaptation engine identified the TPACK component with the lowest current BKT mastery estimate as the personalization target for the following session. That target, together with the participant's current CK mastery level and the week's course topic, was inserted into a structured GPT-4o prompt template that specified a pedagogical framing level: none for CK-only tasks, partial for TK or PK targets, and full for PCK targets. Full framing required the generated task to include an explicit learner-perspective element, such as annotating a solution for a hypothetical secondary school student or diagnosing a described misconception. The adaptation engine employed the GPT-4o API alongside a well-structured prompt template which identified the target TPACK component, the participant's current content knowledge mastery level as estimated by BKT, the course topic for the week, and a pedagogical framing level (none, partial, or full) that was decided by a weakest-component targeting policy derived from the AES adaptation model layer.In the experimental group, participants received three task options at the start of each session: one targeting their weakest TPACK component, one at their current content knowledge level without pedagogical framing, and one of their own choice. A brief explanation of why each option was recommended accompanied the choices. In the control group, participants received the same set of tasks used in the experimental group but delivered in a fixed sequence identical for all control participants, without personalized recommendations.

Instruments

Programming skill was assessed using a 40-item practical assessment covering Python fundamentals. They include: variables, conditionals, loops, functions, lists, dictionaries, recursion, and object-oriented basics, adapted from CodeWorkout and validated in a prior administration with 84 pre-service informatics teachers in Kazakhstan (Edwards & Murali, 2017). Scores are reported as percentages of maximum possible score. TPACK was assessed using the 28-item TPACK Survey for Computing Education adapted by Schmidt et al. and subsequently validated by Saeli et al. for informatics teacher contexts, measuring all seven TPACK components on a five-point Likert scale (Saeli et al., 2011; Schmidt et al., 2009). System usability in the experimental condition was assessed using the System Usability Scale (SUS), a validated ten-item instrument scored on a 0–100 scale (Brooke, 1996). SUS was administered only to experimental participants because control participants did not interact with the personalization interface and therefore had no basis for usability judgement. Adjective ratings follow the interpretation framework of Bangor, Kortum, and Miller (2008). Semi-structured interviews were conducted with a purposive subsample of 12 participants after the study concluded. Sessions were conducted in Kazakh or Russian according to participant preference and audio-recorded with consent. Recordings were transcribed verbatim and translated into English by a

certified bilingual translator; a second bilingual researcher independently verified all translations before analysis.

Procedure

The study ran for eight weeks during the spring semester of the 2023-2024 academic year. Each week involved one two-hour structured lab session in the virtual laboratory environment, covering one Python topic according to the course syllabus. All participants in both groups attended the same eight weekly two-hour lab sessions, followed the same course syllabus, and had equal access to instructor office hours. Instructors were briefed on the study protocol and instructed to maintain equivalent engagement across groups. Digital trace data were collected automatically throughout each session. No additional data collection instruments were administered during sessions themselves.

Quantitative analysis used independent-samples t-tests for group comparisons on pre-test scores and analysis of covariance (ANCOVA) for post-test comparisons with pre-test scores as covariates, controlling for baseline differences. Effect sizes are reported as Cohen's d. The statistical significance threshold was set at alpha = 0.05 with Bonferroni correction applied for the seven TPACK subscale comparisons (adjusted alpha = 0.0071). All analyses used R version 4.3.2. Qualitative data were analyzed using reflexive thematic analysis following Braun and Clarke's six-phase procedure (Braun & Clarke, 2006). Two researchers independently coded the interview transcripts; inter-rater reliability on the initial coding was assessed at Cohen's kappa = 0.81. Disagreements were resolved through discussion and the final theme set was reviewed by a third researcher with expertise in educational technology.

Data Privacy and Ethics

All digital trace data, including keystroke logs, compilation event streams, and container system events, were pseudonymized at the point of collection by replacing participant identifiers with randomly generated numeric codes. Raw data were stored on password-protected university servers accessible only to the two lead researchers, retained for the duration of the analysis phase, and deleted following publication. Participants were provided with a written data collection notice at enrollment that enumerated each trace type collected, explained its analytical purpose, and described retention and deletion procedures. Separate written consent was obtained for trace data collection, interview audio recording, and transcript translation.

Results

Programming Skill Development

Table 1 presents pre-test and post-test programming skill scores for both groups. The experimental group improved from a pre-test mean of 41.3% to a post-test mean of 63.7%, a gain of 22.4 percentage points. The control group improved from 42.1% to 56.9%, a gain of 14.8 points. After adjusting for pre-test scores, the between-group difference was statistically significant and of medium-to-large practical magnitude: $F(1, 29) = 7.14$, $p = 0.012$, $\eta^2 = 0.197$, $d = 0.71$. All experimental participants exceeded their pre-test score; in the control group, two participants showed minimal gains and one showed a slight decrease.

Table 1

Pre-test and post-test programming skill scores (% of maximum) by condition

Group	n	Pre-test M (SD)	Post-test M (SD)	Gain	*d*
Experimental (AI-personalized)	17	41.3 (8.7)	63.7 (7.4)	+22.4	0.71*
Control (standard)	15	42.1 (9.2)	56.9 (8.8)	+14.8	—

Note. ANCOVA $F(1, 29) = 7.14$, $p = .012$, $\eta^2 = .197$. *d* = Cohen's d for experimental vs. control gain.

* $p < .05$ (ANCOVA with pre-test covariate).

TPACK Development

Table 2 presents pre-test and post-test TPACK survey scores for the overall scale and the seven subscales across both groups. The experimental group showed significantly greater overall TPACK gains than the control group: $F(1, 29) = 5.42$, $p = 0.031$, $\eta^2 = .157$, $d = 0.58$. The strongest effect was observed on the Pedagogical Content Knowledge subscale ($d = 0.84$, $p = 0.003$ after Bonferroni correction), which carried the lowest baseline scores in both groups and was the primary target of the weakest-component personalization policy. The Technological Knowledge subscale showed the second-largest effect ($d = 0.61$), while the Pedagogical Knowledge subscale showed a moderate but non-significant effect after Bonferroni correction ($d = 0.48$, $p = .021$, adjusted $\alpha = 0.007$). Content Knowledge gains did not differ significantly between groups ($d = 0.31$, $p = 0.14$), consistent with the system's design intent to ensure CK development through course content exposure in both groups while differentially targeting PCK through personalization.

Table 2

TPACK survey scores (1-5 scale) at pre- and post-test by condition.

TPACK Component	Exp Pre	Exp Post	Ctrl Pre	Ctrl Post	Exp Δ	Ctrl Δ	*d*
Overall TPACK	3.21 (0.44)	3.89 (0.38)	3.18 (0.51)	3.51 (0.47)	+0.68	+0.33	0.58*
TK	3.18 (0.52)	3.82 (0.41)	3.22 (0.48)	3.49 (0.44)	+0.64	+0.27	0.61*
PK	3.31 (0.48)	3.79 (0.43)	3.28 (0.53)	3.52 (0.49)	+0.48	+0.24	0.48
CK	3.44 (0.41)	4.02 (0.36)	3.41 (0.46)	3.87 (0.42)	+0.58	+0.46	0.31
PCK	2.94 (0.56)	3.78 (0.47)	2.97 (0.61)	3.28 (0.53)	+0.84	+0.31	0.84**

Note. Five of the seven TPACK subscales are reported. TCK and TPK are omitted because neither was a primary target of the personalization policy, which directed adaptation toward the four components modeled by the BKT chains: CK, TK, PK, and PCK. Pre-registered analysis therefore treated TCK and TPK as secondary outcomes. Both showed small, non-significant between-group differences ($d < 0.22$, $p > .18$) and are not discussed further.

* $p < .05$ (ANCOVA);

** $p < .05$ after Bonferroni correction. Selected subscales shown; full results available from the corresponding author

System Usability and Acceptance

Usability evaluation of the experimental condition (n = 17) produced a mean SUS score of 72.4 (SD = 11.8). This positions the system within the 'Good' adjective rating category as defined by Bangor, Kortum, and Miller. Among individual items, learnability and consistency received the strongest endorsement from participants, with mean ratings of 4.1 and 3.9 respectively. Cognitive overhead presented the most notable weakness. The reverse-scored item addressing mental burden returned a mean of 2.4, indicating that navigating between the task recommendation interface and the programming workspace introduced periodic disruption to participants' workflow. This finding warrants attention in future design iterations. Willingness to adopt the system in subsequent coursework was nonetheless high. Thirteen of the 17 experimental participants, representing 76.5%, expressed agreement or strong agreement with continued use.

Qualitative Findings

Thematic analysis of the 12 interviews produced four themes: (1) perceived accuracy of personalization, (2) emotional responses to AI knowledge assessment, (3) metacognitive activation through pedagogical framing, and (4) design suggestions for transparency and control.

Theme 1 - Perceived accuracy of personalization. Participants in the experimental condition largely viewed the system's programming skill recommendations as appropriate and well-matched to their developmental stage. The perceived difficulty gradient was a recurring theme in qualitative responses. One participant observed that assigned tasks consistently sat just beyond their most recently completed work, describing them as challenging but achievable. This pattern reflects a degree of adaptive calibration that participants found motivating rather than discouraging. Pedagogical content knowledge recommendations were received less favourably. Two participants reported that PCK-targeted tasks felt disconnected from their own sense of where difficulties lay, raising questions about the alignment between classifier output and candidate self-perception. This tension points to a meaningful limitation. The pedagogical reflection classifier, while functional, does not appear to reliably mirror how candidates themselves understand their PCK development. Whether this reflects genuine misalignment in the model's inferences or a gap between actual and perceived competence remains an open question for subsequent investigation.

Theme 2 - Emotional responses to AI knowledge assessment. Several experimental participants reported discomfort when the system surfaced PCK-targeted recommendations, interpreting the suggestion as an implicit signal that their pedagogical knowledge was deficient. One participant described the experience as follows: "When it told me to focus on explaining to a student, I felt like it was saying I did not really know how to teach this. That was hard to accept at first" (P03, experimental). In contrast, two control participants indicated in post-study interviews that they would have welcomed access to such a system, expressing curiosity about how their TPACK profile compared to their own self-assessment. The finding illustrates a consistent duality: transparency in AI-driven knowledge assessment can activate either productive self-reflection or defensive resistance, depending on how the feedback is framed and whether the learner has agency over the response.

Theme 3 - Metacognitive activation through pedagogical framing. A consistent finding across experimental participants was that PCK-targeted tasks prompted a qualitatively different kind of engagement with the programming material. Participants to annotate solutions with explanations for hypothetical learners or to identify common misconceptions about the topic because tasks required it. Participants described becoming more aware of 'what was hard about this for me when I first learned it,' 'which parts I could not explain clearly,' and 'why someone might write the wrong solution.' One participant articulated this shift directly: "I started thinking about why this would confuse a student who had never seen loops before. I had not thought about it that way when I was learning it myself" (P11, experimental). This metacognitive activation was uniformly perceived positively. It was also cited by all six experimental interview participants as the aspect of the system they valued most.

Theme 4 - Design suggestions for transparency and control. All 12 interview participants, regardless of condition, identified greater transparency and learner control as the most important improvements for future system iterations. Specific suggestions included displaying each participant's TPACK profile graph; allowing participants to dispute or override a task recommendation with an explanation; and providing a brief account of which specific behavior in the previous session triggered the current recommendation. These suggestions are directly addressable through interface design modifications and are incorporated into the design implications discussed in Section 5.

Discussion

The significant positive effects of AI personalization on both programming skill (d = 0.71) and overall TPACK development (d = 0.58) are encouraging for a first empirical evaluation of a novel system. The particularly strong effect on the PCK subscale (d = 0.84) is especially noteworthy. PCK is consistently identified in the teacher education literature as the most difficult TPACK component to develop and the one most poorly served by conventional instruction (Saeli et al., 2011; Mouza et

al., 2021). That the weakest-component targeting policy directed the majority of recommendations toward PCK, the dimension carrying the lowest baseline scores across both groups, and that this focus produced the largest observed effect, together constitute strong evidence that the TPACK-aligned adaptation mechanism functioned as designed. The system identified the most underdeveloped knowledge dimension, concentrated instructional effort there, and the outcome data reflected that priority. This is precisely the pattern an adaptive mechanism of this kind should produce.

The non-significant difference in CK gains between groups deserves careful interpretation. CK development fell outside the system's design scope. Its purpose was to layer PCK and TK growth on top of whatever content knowledge gains the standard curriculum already produced. The outcome pattern reflects this logic. Comparable CK gains across both groups confirm the system neither enhanced nor disrupted the content knowledge trajectory that coursework alone provides, while substantially larger PCK gains in the experimental group indicate that pedagogical development was added without displacing foundational knowledge acquisition. The intervention operated additively, targeting a gap standard instruction leaves unaddressed. This is a key concern in pre-service teacher preparation where instructional time is constrained.

The qualitative findings converge on a central tension in AI-personalized teacher education. The same transparency that makes personalization educationally valuable can trigger defensive emotional responses when applied to professional identity-relevant knowledge like PCK. This tension is not unique to the present system. It parallels findings from formative assessment research showing that feedback on higher-order thinking skills is more threatening to learner identity than feedback on surface-level correctness (Baker & Hawn, 2022). For the adaptive system architecture, this implies that TPACK profile communication must be designed with careful attention to framing: emphasizing growth and development potential rather than current deficiency, providing behavioral evidence for assessments, and enabling learner agency through override mechanisms.

The Theme 3 finding is theoretically significant. PCK-targeted tasks activated qualitatively richer metacognitive engagement than standard tasks. This suggests that the adaptation engine's pedagogical framing mechanism does not merely append an explanatory exercise to an existing programming task. It fundamentally shifts the learner's cognitive orientation toward the material, prompting a dual-perspective engagement as both programmer and future teacher. This cognitive reorientation may be the distinctive mechanism through which PCK development is accelerated. Future research should examine whether this effect is specific to tasks that explicitly require learner-perspective framing or whether other pedagogical designs can achieve it

Three limitations of the present study require acknowledgment. First, the sample size of 32 participants is small, and the quasi-experimental design precludes definitive causal inference about the system's effects. The effect size estimates, while encouraging, carry wide confidence intervals (programming $d = 0.71$, 95% CI [0.04, 1.36]; TPACK $d = 0.58$, 95% CI [-0.04, 1.18]) that overlap with zero at the lower bound, underscoring the preliminary nature of the findings. Second, the study was conducted at a single point in time in a specific institutional and cultural context. Additionally, because experimental and control participants were enrolled at two different universities, institutional factors such as instructor personality, campus learning culture, and local curriculum emphasis cannot be fully disentangled from the intervention effect. A within-institution randomized design in future work would eliminate this confound. Generalization to other teacher education programs, particularly in settings with different pedagogical traditions regarding AI use in education, requires caution. Third, the pedagogical reflection classifier was developed and validated specifically for this study context; its performance in other linguistic and institutional settings is unknown.

Conclusion

This pilot study provides preliminary empirical evidence that TPACK-aligned AI personalization based on digital trace analysis from a containerized programming laboratory can support the dual development objectives of pre-service informatics teacher preparation. The findings are exploratory in nature and should be interpreted with appropriate caution given the small sample, quasi-experimental design, and single-context setting. Deploying a four-component AES architecture with

TPACK-aligned learner modeling produced measurable educational value in an authentic teacher education context. Significantly greater gains in both programming skill and TPACK development among the experimental group, with a particularly pronounced effect on the PCK subscale, confirm that principled adaptive design translates into observable learning outcomes. Qualitative findings present a more nuanced picture. Participants valued the metacognitive activation that pedagogically framed tasks produced and reported high system usability, yet expressed concern about AI-driven PCK assessment and called for greater transparency and learner agency. These concerns are addressable through targeted interface modifications prior to broader deployment. Future work should prioritize a larger pre-registered randomized controlled trial to establish more precise effect size estimates and test robustness across cohort contexts. A longitudinal study following teacher candidates through their student teaching placements would establish whether the PCK gains observed in the virtual laboratory transfer to authentic classroom instruction. The metacognitive activation mechanism identified in Theme 3 also deserves dedicated investigation: understanding the conditions under which dual-perspective pedagogical framing activates this effect would provide a theoretically grounded basis for optimizing the adaptation engine's task generation policy. Together, these studies would build the evidence base needed to responsibly scale this AES approach from a promising pilot to a validated component of pre-service informatics teacher preparation.

References

Aleven, V., Roll, I., McLaren, B. M., & Koedinger, K. R. (2016). Help helps, but only so much: Research on help seeking with intelligent tutoring systems. *International Journal of Artificial Intelligence in Education, 26*(1), 205–223. https://doi.org/10.1007/s40593-015-0089-1

Azcona, D., Hsiao, I.-H., & Smeaton, A. F. (2019). Detecting students-at-risk in computer programming classes with learning analytics from students' digital footprints. *User Modeling and User-Adapted Interaction, 29*(4), 775–806. https://doi.org/10.1007/s11257-019-09234-7

Baker, R. S., & Hawn, A. (2022). Algorithmic bias in education. *International Journal of Artificial Intelligence in Education, 32*(4), 1052–1092. https://doi.org/10.1007/s40593-021-00285-9

Bangor, A., Kortum, P. T., & Miller, J. T. (2008). An Empirical Evaluation of the System Usability Scale. *International Journal of Human–Computer Interaction, 24*(6), 574–594. https://doi.org/10.1080/10447310802205776

Braun, V., & Clarke, V. (2006). Using thematic analysis in psychology. *Qualitative Research in Psychology, 3*(2), 77–101. https://doi.org/10.1191/1478088706qp063oa

Brusilovsky, P., & Peylo, C. (2003). Adaptive and intelligent web-based educational systems. *International Journal of Artificial Intelligence in Education, 13*(2–4), 159–172.

Corbett, A. T., & Anderson, J. R. (1994). Knowledge tracing: Modeling the acquisition of procedural knowledge. *User Modeling and User-Adapted Interaction, 4*(4), 253–278. https://doi.org/10.1007/BF01099821

Denny, P., Prather, J., Becker, B. A., Finnie-Ansley, J., Hellas, A., Leinonen, J., & Sarsa, S. (2024). Computing education in the era of generative AI. *Communications of the ACM, 67*(7), 56–67. https://doi.org/10.1145/3624720

Edwards, S. H., & Murali, K. P. (2017). CodeWorkout: Short programming exercises with built-in data collection. *Proceedings of the 2017 ACM Conference on Innovation and Technology in Computer Science Education, 155–160*. https://doi.org/10.1145/3059009.3059025

Ihantola, P., Vihavainen, A., Ahadi, A., Butler, M., Borstler, J., Edwards, S. H., & Hilton, A. (2015). Educational data mining and learning analytics in programming: Literature review and case studies. *Proceedings of the 2015 ITiCSE on Working Group Reports, 41–63*. https://doi.org/10.1145/2858796.2858798

Ministry of Digital Development, Innovation and Aerospace Industry of the Republic of Kazakhstan. (2017). Digital Kazakhstan state programme 2018–2022. Government of Kazakhstan.

Mouza, C., Yadav, A., & Ottenbreit-Leftwich, A. (Eds.). (2021). Preparing pre-service teachers to teach computer science: Models, practices, and policies. *Information Age Publishing.*

Prather, J., Reeves, B. N., Denny, P., Becker, B. A., Leinonen, J., Luxton-Reilly, A., MacNeil, S., Sarsa, S., Sheard, J., & Tran, A. (2023). The robots are here: Navigating the generative AI revolution in computing education. *Proceedings of the 2023 ITiCSE Working Group Reports, 108–159*. https://doi.org/10.1145/3623769.3623809

Rivers, K., & Koedinger, K. R. (2017). Data-driven hint generation in vast solution spaces: A self-improving Python programming tutor. *International Journal of Artificial Intelligence in Education, 27*(1), 37–64. https://doi.org/10.1007/s40593-015-0070-z

Saeli, M., Perrenet, J., Jochems, W. M. G., & Zwaneveld, B. (2011). Teaching programming in secondary school: A pedagogical content knowledge perspective. *Informatics in Education, 10*(1), 73–88.

Schmidt, D. A., Baran, E., Thompson, A. D., Mishra, P., Koehler, M. J., & Shin, T. S. (2009). Technological pedagogical content knowledge (TPACK): The development and validation of an assessment instrument for preservice teachers. *Journal of Research on Technology in Education, 42*(2), 123–149. https://doi.org/10.1080/15391523.2009.10782544

Standl, B., & Brinda, T. (2020). Design and evaluation-concept for teaching and learning laboratories in informatics teacher education. In Informatics in schools: Engaging learners in computational thinking. ISSEP 2020 (Lecture Notes in Computer Science, Vol. 12518, pp. 118–130). Springer. https://doi.org/10.1007/978-3-030-63212-0_11

Villegas-Ch, W., Garcia-Ortiz, J., & Sanchez-Viteri, S. (2022). Artificial intelligence and learning analytics in teacher education: A systematic review. *Education Sciences, 12*(8), Article 569. https://doi.org/10.3390/educsci12080569

Yadav, A., Stephenson, C., & Hong, H. (2017). Computational thinking for teacher education. *Communications of the ACM, 60*(4), 55–62. https://doi.org/10.1145/2994591

Zhang, Z., Shi, Y., & Huang, R. (2023). Utilizing programming traces to explore and model the dimensions of novices' code-writing skill. *Computer Applications in Engineering Education, 31*(4), 897–912. https://doi.org/10.1002/cae.22622

Гаухар Дуйсенова[1*], *Нуржан Шындалиев*[2], *Рустам Шадиев*[3]
[1,2]Л.Н.Гумилев атындағы Еуразия ұлттық университеті, Астана, Қазақстан
[3]Чжэцзян университеті, Ханчжоу, Қытай
[*]e-mail: gauhar.duisen9@gmail.com

ЦИФРЛЫҚ ІЗДЕРДІ ТАЛДАУ НЕГІЗІНДЕ ЖАСАНДЫ ИНТЕЛЛЕКТ АРҚЫЛЫ ЖЕКЕЛЕНДІРІЛГЕН БАҒДАРЛАМАЛАУ ТАПСЫРМАЛАРЫ: КОНТЕЙНЕРЛЕНГЕН ЗЕРТХАНАЛЫҚ ОРТАДАҒЫ БОЛАШАҚ ИНФОРМАТИКА МҰҒАЛІМДЕРІМЕН ЖҮРГІЗІЛГЕН ПИЛОТТЫҚ ЗЕРТТЕУ

Аңдатпа. Компьютерлік ғылым бойынша болашақ мұғалімдерді даярлау бағдарламалау құзыреттілігі мен педагогикалық мазмұндық білімді бір мезгілде дамытуы тиіс, алайда сандық іздерді талдау негізіндегі жасанды интеллект арқылы тапсырмаларды жекелендіру контейнерлендірілген зертхана ортасында осы екі мақсатты қолдай алатынын бағалайтын эмпирикалық зерттеу жүргізілмеген. Бұл мақала Қазақстандағы екі университеттің екі когортында 32 болашақ информатика мұғалімдерін қамтыған аралас әдісті квазиэкспериментік пилоттық зерттеуді ұсынады. Эксперименттік топтың қатысушылары (n = 17) Docker негізіндегі виртуалды зертханалардағы сандық ізінің нақты уақыттағы талдауы арқылы жасалған AI-персоналдандырылған бағдарламалау тапсырмаларын орындады; бақылау топтары (n = 15) сол ортада ешқандай жекелендірусіз бірдей тапсырмаларды орындады. Сегіз апта бойы эксперименттік топ бағдарламалау дағдыларын бағалау бойынша (d = 0,71, p = 0,012) және расталған TPACK сауалнама құралдары бойынша (d = 0,58, p = 0,031) статистикалық тұрғыдан маңызды жоғары нәтижелерге қол жеткізді, әсіресе оқыту мазмұны

бойынша білім субшкаласында (d = 0,84) әсері айтарлықтай күшті болды. Зерттеуден кейінгі сұхбаттардың сапалық талдауы қатысушылардың тапсырма ұсыныстарының айқындылығын жоғары бағалағанын көрсетті, бірақ олар алгоритмдік әділдік пен жүйенің педагогикалық білімдерін бағалаудағы дәлдігіне қатысты алаңдаушылық білдірді. Бұл нәтижелер сандық іздер негізінде TPACK қағидаларына сәйкес жасалған жасанды интеллект арқылы жекелендірудің болашақ информатика мұғалімдерін даярлауда мүмкін әрі тиімді екеніне алғашқы эмпирикалық дәлелдер ұсынады және кейінгі кезеңдерге нақты дизайн жақсартуларын анықтайды.

Түйін сөздер: жасанды интеллект арқылы жекелендіру, цифрлық іздерді талдау, болашақ мұғалімдер, бағдарламалау білім беруі, контейнерленген виртуалды зертхана, TPACK, аралас әдістерге негізделген пилоттық зерттеу, оқу аналитикасы, Қазақстан

Гаухар Дуйсенова[1], Нуржан Шындалиев[2], Рустам Шадиев[3]*
[1,2] Евразийский национальный университет имени Л.Н. Гумилева, Астана, Казахстан
[3]Чжэцзянский университет, Ханчжоу, Китай
e-mail: gauhar.duisen9@gmail.com

ПЕРСОНАЛИЗИРОВАННЫЕ ЗАДАНИЯ ПО ПРОГРАММИРОВАНИЮ НА ОСНОВЕ ИСКУССТВЕННОГО ИНТЕЛЛЕКТА И АНАЛИЗА ЦИФРОВЫХ СЛЕДОВ: ПИЛОТНОЕ ИССЛЕДОВАНИЕ С БУДУЩИМИ УЧИТЕЛЯМИ ИНФОРМАТИКИ В КОНТЕЙНЕРИЗИРОВАННОЙ ЛАБОРАТОРНОЙ СРЕДЕ

Аннотация. Подготовка будущих учителей информатики должна одновременно развивать навыки программирования и предметные педагогические знания, однако до сих пор не было проведено ни одного эмпирического исследования, в котором бы оценивалось, может ли персонализация заданий с помощью ИИ на основе анализа цифровых следов способствовать достижению этой двойной цели в контейнерной лабораторной среде. В данной статье представлено квазиэкспериментальное пилотное исследование с использованием смешанных методов, в котором приняли участие 32 будущих учителя информатики из двух университетских групп в Казахстане. Участники экспериментальной группы (n = 17) получали персонализированные с помощью ИИ задачи по программированию, сгенерированные на основе анализа их цифровых следов в реальном времени в виртуальных лабораториях на базе Docker; участники контрольной группы (n = 15) выполняли те же задачи в той же среде без индивидуальной настройки. За восемь недель экспериментальная группа достигла статистически значимо более высоких результатов как по оценке навыков программирования (d = 0,71, p = 0,012), так и по валидированным инструментам опроса TPACK (d = 0,58, p = 0,031), причем особенно сильный эффект наблюдался в подшкалах знаний по предмету преподавания (d = 0,84). Качественный анализ интервью, проведенных после завершения исследования, показал, что участники высоко оценили прозрачность рекомендаций по заданиям, но выразили озабоченность по поводу алгоритмической справедливости и точности оценки системой их педагогических знаний. Эти результаты предоставляют первые эмпирические данные, свидетельствующие о том, что персонализация на основе ИИ с учетом модели TPACK, основанная на анализе цифровых следов, является осуществимой и эффективной для подготовки будущих учителей информатики, а также позволяют определить конкретные направления совершенствования методики для последующих итераций.

Ключевые слова: персонализация с использованием искусственного интеллекта, анализ цифровых следов, будущие учителя, обучение программированию, контейнеризированная виртуальная лаборатория, TPACK, пилотное исследование со смешанными методами, аналитика обучения, Казахстан

Received 1 April 2026