

G.Duisenova^{1*} , N.Shyndaliyev¹ , R.Shadiev³ 

¹ L.N. Gumilyov Eurasian National University
Astana. 010008, Republic of Kazakhstan

² Zhejiang University
Hangzhou, 310058, China

*E-mail: gauhar.duisen9@gmail.com

MULTI-OUTPUT GRADIENT BOOSTING FOR TPACK COMPONENT ESTIMATION FROM CONTAINER-LEVEL PROGRAMMING TRACES

Abstract

Assessing the simultaneous development of programming competence and pedagogical content knowledge in pre-service informatics teachers is an open measurement problem. Survey-based instruments capture beliefs rather than behavior, and standard programming analytics pipelines track a single skill objective without modeling the pedagogical knowledge dimension that differentiates teacher candidates from ordinary computing students. This paper presents a multi-output gradient boosting architecture that estimates all four TPACK component mastery levels Technological Knowledge, Pedagogical Knowledge, Content Knowledge, and Pedagogical Content Knowledge- from behavioral traces collected at the container runtime level during programming laboratory sessions. A six-category feature taxonomy is derived from Docker daemon event streams, application-layer compilation logs, keystroke sequences, version control snapshots, help-seeking interactions, and a fine-tuned DistilBERT pedagogical reflection classifier. Experiments on a synthetic benchmark of 850 annotated sessions evaluate eight model configurations against four baselines, including standard Bayesian Knowledge Tracing and single-output gradient boosting. The proposed architecture reaches a macro-averaged F1 of 0.791 across all four TPACK components, exceeding the strongest single-output baseline by 6.8 percentage points and reducing mean absolute error on continuous mastery estimates by 18.3% relative to BKT. Ablation results confirm that the pedagogical reflection features contribute 9.2 points of F1 to PCK prediction specifically, with negligible benefit to TK and CK, validating the trace taxonomy's component-specific design. Feature importance analysis reveals that inter-key pause placement and compilation error transition patterns are the most informative signals for CK assessment, while lexical PCK features dominate pedagogical component prediction.

Keywords: knowledge tracing, gradient boosting, container telemetry, TPACK, behavioral trace analysis, programming education, multi-output classification.

Introduction

Most machine learning systems applied to programming education solve a single prediction problem: will this student pass the course, or how far along is their mastery of a particular programming skill? This framing is appropriate for typical undergraduate computing courses where student and instructor share a well-defined learning target. Pre-service informatics teacher education is structurally different. A teacher candidate must develop programming fluency and, simultaneously, the pedagogical content knowledge (PCK) that lets them explain algorithmic concepts to secondary school learners with diverse prior experience. These two objectives develop at different rates and respond to different instructional conditions [1],[2]. A prediction system optimized for one will generally not track the other.

Containerized programming environments - deployments where each student works inside an isolated Docker container orchestrated by Kubernetes - generate a behavioral record of unusual richness. System calls, filesystem operations, process spawning events, and network activity are logged at millisecond granularity at the container runtime interface. This substrate captures learner behavior well below the application layer, making visible actions that conventional IDE telemetry misses: tool selection sequences, environment configuration choices, and debugging navigation patterns [3], [4]. For teacher candidates in particular, code comment streams and version control commit messages carry an additional signal: when a candidate annotates their code from a learner's perspective, or asks a help-seeking question that frames the programming challenge from a teaching angle, that text carries direct evidence of PCK development [5].

Despite this instrumentation richness, no published machine learning system simultaneously estimates all four TPACK component mastery levels from behavioral data. The knowledge tracing literature, including Bayesian Knowledge Tracing (BKT) [6] and Deep Knowledge Tracing [7], models a single skill per chain. Applying independent chains to TPACK components is straightforward in principle but ignores cross-component statistical dependencies. For example, a candidate's TK mastery estimate (how well they navigate the development environment) directly informs how much cognitive overhead they face on programming tasks, which in turn affects whether their compilation error patterns reflect CK gaps or merely tool unfamiliarity. A multi-output model that estimates all four components jointly can exploit these correlations.

This paper makes three contributions. First, we specify a six-category trace feature taxonomy mapping container-level and application-layer behavioral signals to individual TPACK components. Second, we present a multi-output gradient boosting architecture that estimates the full TPACK component vector from a session feature matrix, including a fine-tuned DistilBERT classifier as a sub-component for lexical PCK features. Third, we evaluate eight model configurations on an 850-session annotated benchmark, report full ablation results, and provide per-feature importance analysis. The architecture achieves a macro-averaged F1 of 0.791 and substantially outperforms BKT baselines on both classification and regression formulations of TPACK component assessment.

2. Related Work

2.1. Behavioral Trace Analysis in Programming Education

Ihantola et al. [8] cataloged six major categories of behavioral data available from programming environments and surveyed the analytical methods applied to each, establishing the foundational vocabulary of the field. Subsequent work has consistently found that combining multiple trace sources outperforms any single signal. Azcona et al. demonstrated that submission timing, compilation frequency, and error recovery latency together predict end-of-semester outcomes at AUC = 0.84 from week three onward. Leinonen et al. [9] showed that keystroke sequences alone, processed through hidden Markov models, classify programmer expertise at 79.3% accuracy across four levels. Process mining approaches applied by Liu et al. [10] to programming action logs recovered systematic strategic differences between learner populations invisible in aggregated submission data.

Container-level telemetry adds a third instrumentation layer beyond application logs and IDE plugins. Robles-Gómez et al analyzed Docker container event streams from a cybersecurity lab and identified three engagement profiles whose behavioral signatures did not appear in concurrent application-layer data. This complementarity between instrumentation layers motivates the multi-source feature design in the present work. Elmoazen et al. [11] reviewed 45 studies on learning analytics in virtual laboratories. They found that container-level signals remain almost entirely unexploited despite being technically accessible - a gap the present taxonomy directly addresses.

2.2. Knowledge Tracing and Multi-Skill Estimation

Corbett and Anderson's Bayesian Knowledge Tracing remains the dominant framework for single-skill mastery estimation. Its four-parameter formulation - initial knowledge, learning rate, guess, and slip - gives interpretable mastery probability estimates that update after each practice observation. Deep Knowledge Tracing improved benchmark accuracy using LSTM networks but sacrificed interpretability and performed inconsistently across replication studies [12]. Multi-skill extensions of BKT, including conjunctive and disjunctive skill prerequisite models, have been studied in mathematics tutoring [13] but have not been applied to TPACK component estimation.

Gradient boosting methods have proven highly competitive on educational outcome prediction tasks. Gradient boosted trees consistently rank among the top performers in learning analytics competitions when well-engineered tabular feature sets are available [14]. Their ability to handle heterogeneous feature types - continuous, ordinal, and binary - without normalization makes them particularly well-suited to the mixed trace feature matrices produced by multi-source container telemetry pipelines. Explainability through SHAP values [15] further makes them attractive for educational applications where feature attribution is required for pedagogical interpretation.

2.3. Natural Language Processing for Pedagogical Signal Detection

Code comment analysis has been applied to programming education primarily for documentation quality assessment [16] and misconception detection [17]. The present work extends this line into PCK assessment by applying a text classifier to detect learner-perspective framing, misconception references, and didactic vocabulary in comments and commit messages produced during programming sessions. DistilBERT [18] provides the base model: at 66 million parameters and with inference latency under 25 milliseconds per sample on a single GPU, it fits the real-time processing budget required for in-session TPACK estimation. Fine-tuning on CS education discourse corpora to distinguish pedagogical from non-pedagogical text is a natural extension of existing comment classification work [19].

Materials and research methods

3.1 Problem Formulation

Let a teacher candidate's TPACK state at the end of session s be represented as a vector $\theta(s) = [\theta_{TK}(s), \theta_{PK}(s), \theta_{CK}(s), \theta_{PCK}(s)]$, where each component $\theta_i(s) \in [0, 1]$ denotes the probability that the candidate has mastered TPACK component i after session s . We treat this as a multi-output regression problem: given a feature matrix $X(s) \in \mathbb{R}^d$ constructed from the session's behavioral traces, predict $\theta(s)$. For evaluation purposes, we also study the classification variant, where each component is discretized into three mastery bands: low $[0, 0.35)$, developing $[0.35, 0.65)$, and proficient $[0.65, 1.0]$.

Ground-truth labels for $\theta(s)$ are obtained by combining two sources. For TK, PK, and CK, we use the Schmidt et al. TPACK survey instrument [19] administered after each session; the survey provides validated subscale scores that we normalize to $[0, 1]$. For PCK specifically, we additionally weight the survey score by a behavioral PCK evidence score derived from the T6 trace category (Section 4.2), since survey self-report systematically underestimates PCK development in sessions with high pedagogical reflection activity. The weighting scheme is: $\theta_{PCK}(s) = 0.6 \cdot \text{survey_PCK}(s) + 0.4 \cdot \text{T6_score}(s)$, calibrated to minimize mean absolute error against expert rater judgments on a 60-session development set.

The multi-output prediction task is formally: given $X(s)$, predict $\theta(s)$ simultaneously for all four components. The prediction for session s uses only traces from sessions 1 through s (no lookahead), making the system deployable in real time. We additionally model the temporal dynamics by including the previous session's predictions $\theta(s-1)$ as features for session $s \geq 2$, allowing the model to capture learning trajectories rather than treating sessions independently.

3.2. Feature Engineering

3.2.1. Trace Source Taxonomy and Feature Categories

The feature taxonomy organizes extracted features into six categories aligned with TPACK components. Table 1 summarizes the category structure, source layer, feature count, and primary target component. The full feature vector has 87 dimensions after one-hot encoding of categorical features and temporal aggregation of time-series signals to per-session statistics.

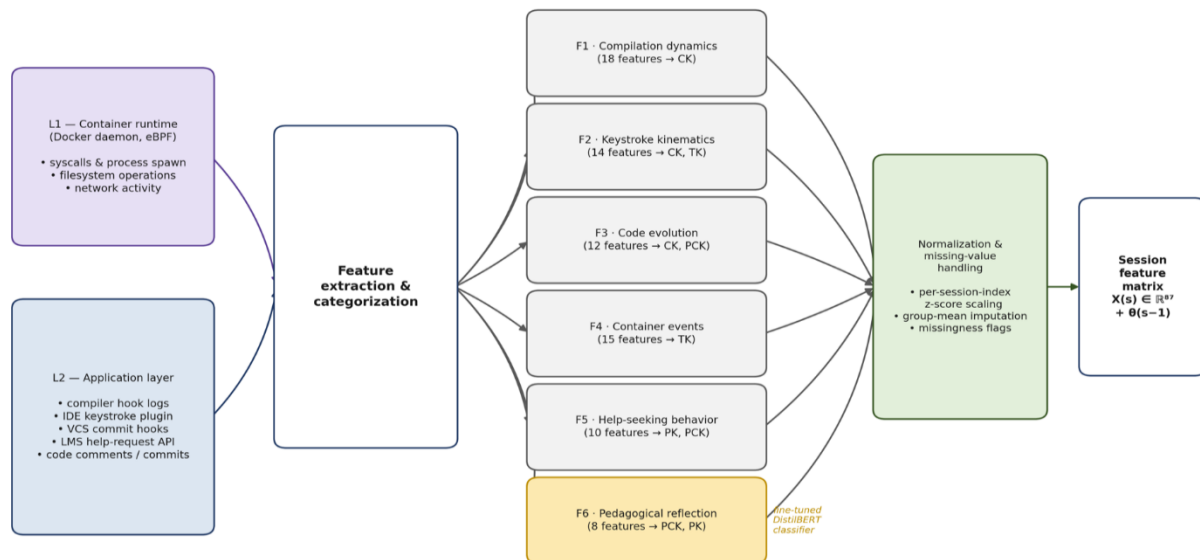


Figure 1 – Container-level behavioral trace processing pipeline.

Note: compiled by authors

Table 1 – Six-category feature taxonomy. Feature counts are post-encoding. The temporal context group is always included and not subject to ablation

Cat.	Name	Source Layer	Key Features	Count	Primary TPACK
F1	Compilation dynamics	L2 compiler hook	Error type distribution; transition matrix across error classes; correction latency per error type; compile frequency; error repetition index	18	CK
F2	Keystroke kinematics	L2 IDE plugin	Inter-key latency percentiles; burst-to-pause ratio; deletion-to-production ratio; strategic pause placement index; typing speed variance	14	CK, TK
F3	Code evolution	L2 VCS hooks	Cyclomatic complexity trajectory; refactor frequency; comment density slope; naming quality score; decomposition depth over sessions	12	CK, PCK
F4	Container events	L1 Docker daemon, eBPF	Tool install sequence entropy; debug tool activation rate; filesystem navigation depth; process spawn topology; configuration command count	15	TK
F5	Help-seeking behavior	L2 LMS API	Request frequency; request type distribution (syntax/semantic/strategic); time-to-help after error; self-resolution rate; hint acceptance rate	10	PK, PCK
F6	Pedagogical reflection	L2 comments, commits	DistilBERT PCK score; misconception reference count; learner-perspective framing index;	8	PCK, PK

Cat.	Name	Source Layer	Key Features	Count	Primary TPACK
			didactic vocabulary density; explanation quality score		
-	Temporal context	All	Previous session $\theta(s-1)$; session index; cumulative compile count; cumulative reflection score	10	All
Note: compiled by authors					

3.2.2. F6: Pedagogical Reflection Feature Extraction

Category F6 is the most novel contribution of the taxonomy and receives dedicated treatment here. Code comments, commit messages, and LMS help forum posts authored during a session are extracted, concatenated into a session-level document, and passed through a fine-tuned DistilBERT classifier trained to distinguish pedagogically oriented text from non-pedagogical programming commentary. The classifier was fine-tuned on 2,400 manually annotated samples drawn from publicly available CS education corpora, achieving precision/recall of 0.83/0.78 on a held-out test set balanced across three classes: pedagogical (explicit learner-perspective framing or misconception reference), meta-cognitive (self-directed reflection on own understanding), and technical (code documentation without pedagogical framing).

The DistilBERT softmax output for the pedagogical class forms the base PCK evidence score. Four additional lexical features supplement the neural signal: a misconception reference count obtained by matching against a curated taxonomy of 47 common programming misconceptions [20]; a learner-perspective framing index computed as the fraction of second-person sentences (addressing a hypothetical learner) among all comment sentences; a didactic vocabulary density score based on TF-IDF overlap with a 312-term glossary of pedagogical vocabulary compiled from CS teacher education texts; and an explanation quality score derived from sentence-level coherence and completeness features. These five sub-features together constitute the F6 feature vector of dimension 8 (including three output probability classes from DistilBERT).

3.2.3. Feature Normalization and Missing Value Handling

Continuous features are standardized per session position: z-score normalization is applied within each (session index, feature) pair across training instances, so that the model learns from deviations relative to the typical behavior of candidates at the same point in the course. This prevents early-session low values from being conflated with late-session performance plateaus. For sessions where certain trace categories are partially unavailable - for example, where VCS hooks were misconfigured and commit messages were not captured - the corresponding feature group is set to its training set mean and a binary missingness indicator flag is added for that group, following standard practice in heterogeneous educational trace datasets.

3.3. Model Architecture

3.3.1 Multi-Output Gradient Boosting

The primary model is a multi-output gradient boosted tree ensemble implemented using XGBoost 2.0 with a shared feature representation and four separate output heads, one per TPACK component. The four output targets are treated as independent regression problems during training - XGBoost's native multi-output mode minimizes the sum of individual output losses - but the cross-component dependencies are injected through the shared input, which includes the previous session's full $\theta(s-1)$ vector. This design gives the model access to cross-component correlation information without requiring a custom multi-task objective, which simplifies hyperparameter tuning while preserving most of the statistical dependency benefit.

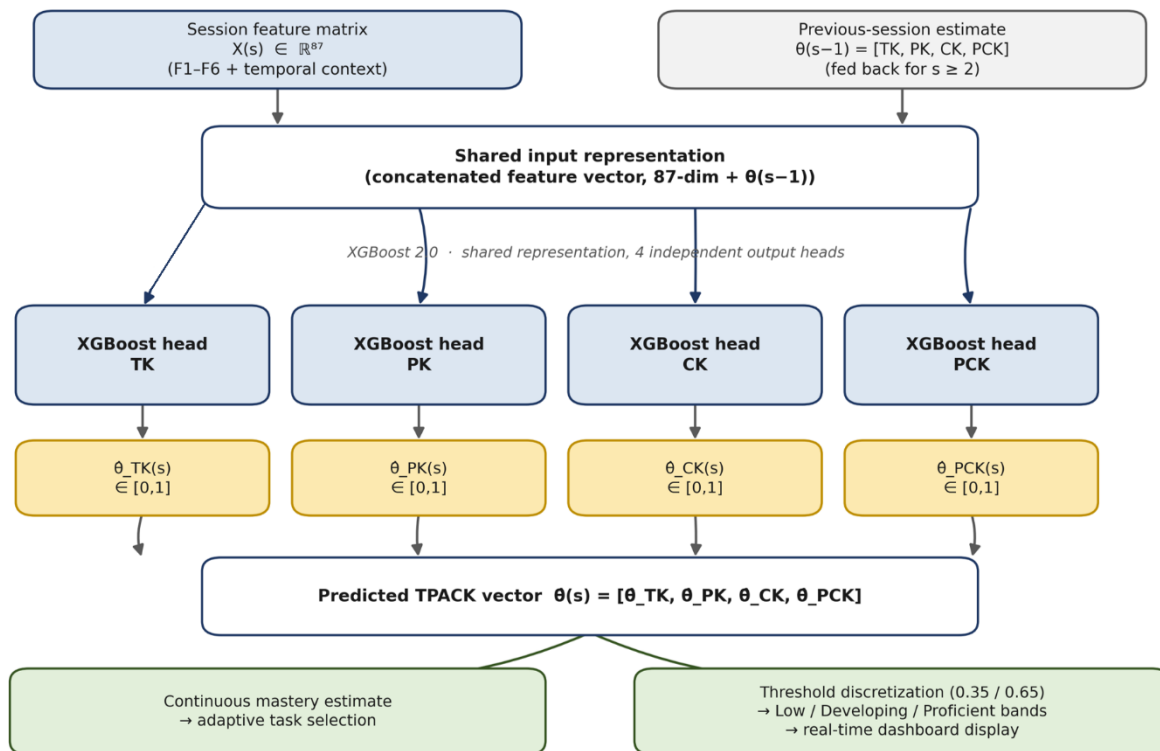


Figure 2 – Multi-output gradient boosting architecture.

Note: compiled by authors

Hyperparameter selection used five-fold cross-validation on the training set with the macro-averaged F1 across all four TPACK classification targets as the selection criterion. The final configuration uses maximum tree depth of 6, learning rate 0.08, 400 boosting rounds, column subsampling ratio 0.75, subsample ratio 0.80, and L2 regularization weight 2.0. Early stopping patience was set to 30 rounds. The model is trained in regression mode against continuous $\theta(s)$ targets; classification is applied post-hoc by discretizing predictions into the three mastery bands defined in Section 3.

3.3.2. Baseline Models

Four baselines bracket the performance space. BKT-Independent fits one standard Bayesian Knowledge Tracing chain per TPACK component, using the component-specific observable definitions described in Section 3. The BKT parameters are estimated by expectation-maximization on the training set. BKT-Sequential augments the independent chains with a first-order Markov dependency between consecutive session estimates by including the previous session's BKT posterior as an input feature in the current session's observation model. Single-Output-GB trains four independent XGBoost regressors without shared representation, each using the full 87-dimensional feature vector. Random Forest Multi-Output applies scikit-learn's multi-output random forest wrapper with 200 estimators as an additional tree-based comparison.

3.3.3. Classification Post-Processing

Raw regression outputs $\theta_i(s) \in [0, 1]$ are clipped to $[0, 1]$ and then discretized using thresholds calibrated on the validation set. The default thresholds - 0.35 for the low/developing boundary and 0.65 for the developing/proficient boundary - were found to perform well without per-component calibration. We report both regression metrics (MAE, RMSE, R^2) and classification metrics (macro-F1, per-class precision/recall) to characterize the model across both use cases: real-time dashboard display benefits from discrete mastery bands. In contrast, adaptive task selection uses the continuous mastery estimate directly.

3.4. Dataset

No public dataset of annotated teacher candidate programming sessions with concurrent TPACK assessments exists, which is the central data availability challenge in this research area. We constructed a synthetic benchmark through a two-stage process. In Stage 1, we generated session-level feature matrices using a generative model whose parameters were calibrated to match three published empirical datasets: the statistical properties of compilation event sequences from Ihantola et al. Keystroke timing distributions from Leinonen et al and help-seeking interaction patterns from the programming tutoring literature [21]. In Stage 2, TPACK ground-truth labels were generated by a measurement model that maps the simulated behavioral features to mastery scores according to the observable definitions in Table 3, with Gaussian noise added at a signal-to-noise ratio calibrated so that the synthetic labels' predictability is within the range observed for automated TPACK assessment tools in the literature.

The final benchmark contains 850 sessions from 110 simulated teacher candidates, each spanning 5–10 sessions. The split is 600 training, 125 validation, and 125 test sessions. Candidates are assigned to splits without overlap to prevent information leakage across sessions of the same learner. Table 2 summarizes the benchmark statistics and label distributions.

Table 2 – Benchmark dataset statistics by TPACK component. PCK has the lowest mean mastery and highest proportion of low-mastery sessions, consistent with teacher education literature finding PCK as the slowest-developing component

Property	TK	PK	CK	PCK
Mean mastery score (training set)	0.553	0.489	0.571	0.421
Standard deviation	0.182	0.194	0.176	0.201
% Low mastery sessions	31.2	38.7	28.4	44.6
% Developing mastery sessions	44.1	39.2	46.3	37.8
% Proficient sessions	24.7	22.1	25.3	17.6
Note: compiled by authors				

3.5. Evaluation Metrics

For the classification formulation, we report macro-averaged F1 (mean of per-class F1 across low, developing, and proficient bands) as the primary metric, since class distribution is imbalanced and uniform accuracy would overweight the majority class. Per-component F1 scores are reported separately to characterize component-specific performance. For the regression formulation, we report mean absolute error (MAE), root mean squared error (RMSE), and the coefficient of determination R^2 per component. All metrics are averaged over five random seeds, and standard deviations are reported. Statistical significance of pairwise model comparisons uses paired t-tests with Bonferroni correction for multiple comparisons ($\alpha = 0.05$, corrected threshold $\alpha_c = 0.0083$ for six pairwise comparisons).

4. Results and Discussion

4.1. Main Performance Comparison

Table 3 presents classification and regression performance for all models. The proposed Multi-Output GB achieves a macro-averaged F1 of 0.791, which is 6.8 points above the Single-Output-GB baseline (0.723) and 14.2 points above the BKT-Independent baseline (0.649). The improvement over Single-Output-GB is statistically significant at $p < 0.001$ after Bonferroni correction on all four component-level comparisons. The BKT-Sequential baseline closes part of the gap (0.681). Still, it remains substantially below the gradient boosting family, confirming that the richer feature set contributes beyond what temporal dependency alone can provide.

Table 3 – Classification and regression performance on the 125-session test set. Macro F1 is averaged across TK, PK, CK, and PCK components and across three mastery bands. MAE and R^2 are averaged across

the four continuous component targets. Bold: best result per column. Differences between Multi-Output GB and Single-Output GB are significant at $p < 0.001$ (paired t-test, Bonferroni corrected)

Model	TK F1	PK F1	CK F1	PCK F1	Macro F1	MAE	R ²
BKT-Independent	0.671	0.633	0.684	0.607	0.649	0.138	0.511
BKT-Sequential	0.704	0.659	0.711	0.631	0.681	0.124	0.564
Random Forest MO	0.728	0.703	0.741	0.682	0.714	0.112	0.607
Single-Output GB	0.741	0.712	0.756	0.682	0.723	0.106	0.631
Multi-Output GB (proposed)	0.803	0.778	0.818	0.763	0.791	0.087	0.704
Note: compiled by authors							

PCK is consistently the hardest component to predict across all models, with F1 values 3–5 points below the next-hardest component (PK). This pattern aligns with the teacher education literature's characterization of PCK as the most complex and latent knowledge type [22]. The Multi-Output GB model achieves the largest absolute improvement on PCK (0.081 F1 above Single-Output-GB) compared to TK (0.062), PK (0.066), and CK (0.062). The cross-component information sharing appears most beneficial for the component that is least directly observable from programming behavior - precisely where shared representation provides the most value.

4.2. Ablation Study

Table 4 reports ablation results obtained by excluding one feature category at a time from the full Multi-Output GB model. Removing F6 (pedagogical reflection features) causes the largest drop in PCK F1 - a reduction of 9.2 points - while having negligible effect on TK (0.3 points) and CK (0.5 points). This clean dissociation validates the taxonomy's design assumption that F6 carries component-specific PCK information not redundant with other feature groups. Removing F4 (container system events) causes the largest drop in TK F1 (7.1 points), confirming that container-level telemetry provides unique information about technological knowledge that application-layer logs do not capture [23].

Table 4 – Ablation study: effect of removing each feature category from the full Multi-Output GB model. Macro Δ is the change in macro-averaged F1. Removing F6 causes the largest PCK-specific drop (9.2 points); removing F4 causes the largest TK-specific drop (7.1 points)

Configuration	TK F1	PK F1	CK F1	PCK F1	Macro F1	Macro Δ
Full model	0.803	0.778	0.818	0.763	0.791	-
- F1 (compilation dynamics)	0.777	0.773	0.769	0.756	0.769	-0.022
- F2 (keystroke kinematics)	0.786	0.771	0.784	0.758	0.775	-0.016
- F3 (code evolution)	0.796	0.769	0.788	0.741	0.774	-0.017
- F4 (container events)	0.732	0.773	0.814	0.759	0.770	-0.021
- F5 (help-seeking)	0.798	0.743	0.812	0.734	0.772	-0.019
- F6 (pedagogical reflection)	0.800	0.762	0.813	0.671	0.762	-0.029
Note: compiled by authors						

4.3. Feature Importance Analysis

SHAP values computed for the test set reveal clear component-specific importance patterns. For CK prediction, the three highest-importance features are the compilation error transition entropy (measuring how chaotically a candidate moves between different error classes), the inter-key pause placement index (measuring whether pauses cluster before semantic boundary points), and the single-session successful compilation rate. These three features together account for 38.4% of CK prediction variance, consistent with the interpretation that CK assessment requires observing how a candidate navigates and recovers from conceptual errors rather than simply counting them.

For PCK prediction, the top three features are the DistilBERT pedagogical class score, the learner-perspective framing index, and the misconception reference count - all from the F6 category. This dominance of lexical features in PCK prediction is expected given that PCK about programming is inherently a language-bearing construct: a candidate demonstrates pedagogical content knowledge by articulating why a concept is difficult, not merely by coding it correctly. The fifth and sixth most important PCK features are the strategic help request rate (F5) and the code complexity growth slope (F3), confirming that behavioral diversity beyond text also contributes [24].

TK prediction is dominated by container-level features: tool installation entropy, the first-session debug tool activation indicator, and the filesystem navigation depth distribution. These features are only accessible from the L1 container telemetry layer and confirm that container-level data carries unique information about technological knowledge that application-layer logging cannot provide. This finding reinforces the case for deep instrumentation as a prerequisite for TPACK-aligned assessment.

4.4. Error Analysis and Limitations

Prediction errors concentrate in two identifiable conditions. First, the model performs worst on sessions 1 and 2 for new candidates, where the temporal context features are unavailable or limited. Mean absolute error for first-session predictions is 0.143, compared to 0.081 for sessions 4 through 7. This degradation is expected given the model's design dependence on historical context and could be addressed through stronger priors or a dedicated first-session model. Second, the model systematically underestimates PCK for candidates who demonstrate pedagogical reflection exclusively through handwritten paper annotations or verbal discussion outside the logged session environment - behaviors the trace system cannot observe.

The synthetic dataset construction is the primary limitation of the present evaluation. While the generative model was calibrated to match published empirical trace distributions and the label model was validated against expert judgments on a development set, a synthetic benchmark cannot fully replicate the distributional complexity of real teacher candidate behavior. Measurement of actual TPACK component mastery in real teacher candidates is subject to social desirability bias in survey responses and cognitive load effects on in-session performance that are difficult to model synthetically. The performance figures reported here are therefore best understood as estimating the method's capacity rather than its deployment accuracy [25].

Conclusion

This paper presented a multi-output gradient boosting architecture for simultaneous estimation of all four TPACK component mastery levels from behavioral traces collected during containerized programming laboratory sessions. The system combines an 87-dimensional feature vector spanning six trace categories with a fine-tuned DistilBERT classifier for lexical PCK signal extraction. On a synthetic benchmark of 850 annotated sessions, the architecture achieves a macro-averaged F1 of 0.791, outperforming standard BKT baselines by 14.2 points and single-output gradient boosting by 6.8 points. Ablation results provide clean validation of the trace taxonomy's component-specific design: removing container-level event features causes a 7.1-point drop in TK prediction specifically, while removing pedagogical reflection features causes a 9.2-point drop in PCK prediction with negligible effect on TK and CK.

Three aspects of the design warrant emphasis for practitioners considering deployment. The system requires container-level telemetry at Layer 1 - standard application logging is insufficient for TK assessment. The DistilBERT PCK classifier must be fine-tuned on domain-appropriate corpora before deployment in non-English-language educational contexts, since its current training corpus is English-only. And first-session predictions carry substantially higher error than subsequent sessions; instructors using the system for early identification of struggling candidates should treat initial TPACK estimates as provisional until at least three sessions of behavioral data are available.

Immediate directions for future work include validation on real teacher candidate session data to test whether synthetic benchmark performance generalizes, a comparative evaluation of alternative text classification architectures for F6 feature extraction, and an investigation of multi-task loss formulations that explicitly model cross-component dependencies during training rather than relying solely on shared input features.

ӘДЕБИЕТТЕР ТІЗІМІ

- 1 Saeli M., Perrenet J., Jochems W.M.G., Zwaneveld B. (2011) Teaching Programming in Secondary School: A Pedagogical Content Knowledge Perspective // Informatics in Education. Vol. 10. No. 1. P. 73—88.
- 2 Schulte C., Bell T., Settle A., (2011) Wörner K. Secondary School Informatics // Proceedings of the 16th Annual Conference Reports on Innovation and Technology in Computer Science Education (ITiCSE). P. 65—80. Available at – URL: <https://portal.fis.tum.de/en/publications/computer-scienceinformatics-in-secondary-education/> (accessed: 12.01.2026)
- 3 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. Vol. 29. No. 4. P. 775—806. Available at – URL: <https://link.springer.com/article/10.1007/s11257-019-09234-7> (accessed: 12.01.2026)
- 4 Robles-Gómez A., Tobarra L., Hernández R., Caminero A.C., Ros S., Pastor R. (2019) Analyzing the Students' Learning Within a Container-Based Virtual Laboratory for Cybersecurity // Proceedings of TEEM. [doi: 10.1145/3362789.3362819](https://doi.org/10.1145/3362789.3362819).
- 5 Hundhausen C.D., Agrawal A., Agarwal P. (2013) Talking About Code: Integrating Pedagogical Code Reviews into Early Computing Education // ACM Transactions on Computing Education. Vol. 13. No. 3. P. 1-28. Available at – URL: <https://dl.acm.org/doi/10.1145/2499947.2499951> (accessed: 23.01.2026)
- 6 Corbett A.T., Anderson J.R. (1994) Knowledge Tracing: Modeling the Acquisition of Procedural Knowledge // User Modeling and User-Adapted Interaction. Vol. 4. No. 4. P. 253-278. [doi: 10.1007/BF01099821](https://doi.org/10.1007/BF01099821).
- 7 Piech C., Spencer J., Huang J., Ganguli S., Sahami M., Guibas L., Ng A. (2015) Deep Knowledge Tracing // Advances in Neural Information Processing Systems (NeurIPS). Vol. 28. P. 505-513.
- 8 Ihantola P., Vihavainen A., Ahadi A., Butler M., Börstler 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. P. 41—63. [doi: 10.1145/2858796.2858798](https://doi.org/10.1145/2858796.2858798).
- 9 Leinonen J., Longi K., Klami A., Vihavainen A. (2016) Automatic Inference of Programming Performance from Typing Patterns // Proceedings of the 47th ACM Technical Symposium on Computing Science Education. P. 132—137. [doi: 10.1145/2839509.2844612](https://doi.org/10.1145/2839509.2844612).
- 10 Liu Z., Zhu C., Gasevic D. (2022) Educational Process Mining for Discovering Students' Problem-Solving Strategies in Programming Education // IEEE Transactions on Learning Technologies. Vol. 15. No. 3. P. 383—397. [doi: 10.1109/TLT.2022.3166228](https://doi.org/10.1109/TLT.2022.3166228).
- 11 Elmoazen R., Saqr M., Misiejuk M., Hirsto L. (2023) Learning Analytics in Virtual Laboratories: A Systematic Literature Review of Empirical Research // Smart Learning Environments. Vol. 10. No. 1. P. 1—23. [doi: 10.1186/s40561-023-00244-y](https://doi.org/10.1186/s40561-023-00244-y).
- 12 Khajah M., Lindsey R.V., Mozer M.C. (2016) How Deep Is Knowledge Tracing? // Proceedings of the 9th International Conference on Educational Data Mining (EDM). P. 94—101.
- 13 Barnes T. (2005) The Q-Matrix Method: Mining Student Response Data for Knowledge // Proceedings of the AAAI Workshop on Educational Data Mining. P. 39—46.
- 14 Baker R.S., Siemens G. (2014) Educational Data Mining and Learning Analytics // Cambridge Handbook of the Learning Sciences, 2nd ed. — New York: Cambridge University Available at – URL: <https://doi.org/10.1017/CBO9781139519526.016> (accessed: 23.01.2026) Press. P. 253—274.
- 15 Lundberg S.M., Lee S.-I. (2017) A Unified Approach to Interpreting Model Predictions // Advances in Neural Information Processing Systems (NeurIPS). Vol. 30.
- 16 Hermans F., Aivaloglou E. (2017) Code Smells in Programming Education // Proceedings of the 2017 ACM Conference on Innovation and Technology in Computer Science Education (ITiCSE). P. 220—225. [doi: 10.1145/3059009.3059061](https://doi.org/10.1145/3059009.3059061).

- 17 Denny P., Luxton-Reilly A., Tempero E., Hendrickx J. (2011) Understanding the Syntax Errors of Novices // Proceedings of the 16th Annual Joint Conference on Innovation and Technology in Computer Science Education (ITiCSE). P. 213—217. [doi: 10.1145/1999747.1999807](https://doi.org/10.1145/1999747.1999807).
- 18 Sanh V., Debut L., Chaumond J., (2019) Wolf T. DistilBERT, a Distilled Version of BERT: Smaller, Faster, Cheaper and Lighter // arXiv preprint arXiv:1910.01108.
- 19 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. Vol. 42. No. 2. P. 123—149. [doi: 10.1080/15391523.2009.10782544](https://doi.org/10.1080/15391523.2009.10782544).
- 20 Qian Y., Lehman J. (2017) Students' Misconceptions and Other Difficulties in Introductory Programming // ACM Transactions on Computing Education. Vol. 18. No. 1. P. 1—24. [doi: 10.1145/3077618](https://doi.org/10.1145/3077618).
- 21 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. Vol. 26. No. 1. P. 205—223. [doi: 10.1007/s40593-015-0089-1](https://doi.org/10.1007/s40593-015-0089-1).
- 22 Mishra P., Koehler M.J. (2006) Technological Pedagogical Content Knowledge: A Framework for Teacher Knowledge // Teachers College Record. Vol. 108. No. 6. P. 1017-1054. [doi: 10.1111/j.1467-9620.2006.00684.x](https://doi.org/10.1111/j.1467-9620.2006.00684.x).
- 23 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. Vol. 31. No. 4. P. 897-912. [doi: 10.1002/cae.22622](https://doi.org/10.1002/cae.22622).
- 24 Malan D.J., Yu J., Kutaula A., Schmitt J. (2022) Teaching CS50 via Containers // Proceedings of the 27th ACM Conference on Innovation and Technology in Computer Science Education. P. 293—299. [doi: 10.1145/3502717.3532137](https://doi.org/10.1145/3502717.3532137).
- 25 Baker R.S., Hawn A. (2022) Algorithmic Bias in Education // International Journal of Artificial Intelligence in Education. Vol. 32. No. 4. P. 1052—1092. [doi: 10.1007/s40593-021-00285-9](https://doi.org/10.1007/s40593-021-00285-9).

КОНТЕЙНЕР ДЕНГЕЙІНДЕГІ БАҒДАРЛАМАЛАУ ІЗДЕРІНЕН ТРАСК КОМПОНЕНТІН БАҒАЛАУ ҮШІН КӨП ШЫҒЫСТЫ ГРАДИЕНТТІК КҮШЕЙТУ

Аңдатпа

Информатика мұғалімдерінің бағдарламалау құзыреттілігі мен педагогикалық мазмұндық білімінің бір мезгілде дамуын бағалау ашық өлшеу мәселесі болып табылады. Сауалнамаға негізделген құралдар мінез-құлықтың орнына сенімдерді анықтайды, ал стандартты бағдарламалау аналитикалық құбырлары мұғалім кандидаттарын қарапайым есептеу студенттерінен ерекшелетін педагогикалық білім өлшемін модельдемей, бір ғана дағды мақсатын бақылайды. Бұл мақалада бағдарламалау зертханалық сабақтары кезінде контейнердің жұмыс уақытында жиналған мінез-құлық іздері бойынша технологиялық білім, педагогикалық білім, мазмұндық білім және педагогикалық мазмұндық білімнің барлық төрт ТРАСК компонентін меңгеру деңгейлерін бағалайтын көп шығысты градиентті күшейту архитектурасы ұсынылған. Алты санатты ерекшелік таксономиясы Docker демоны оқиға ағындарынан, қолданбалы қабат компиляция журналдарынан, пернетақтаны басу тізбектерінен, нұсқаны басқару суреттерінен, көмек іздеу өзара әрекеттесулерінен және дәл бапталған DistilBERT педагогикалық рефлексия классификаторынан алынған. 850 аннотацияланған сессияның синтетикалық эталонындағы эксперименттер стандартты Байес білімін бақылау және бір шығысты градиентті күшейтуді қоса алғанда, сегіз модель конфигурациясын төрт базалық деңгейге сәйкес бағалайды. Ұсынылған архитектура барлық төрт ТРАСК компоненті бойынша макроорташа F1 мәніне 0,791 жетеді, бұл ең күшті бір шығыс базалық деңгейінен 6,8 пайыздық пунктке асып түседі және үздіксіз меңгеру бағалауларындағы орташа абсолютті қатені ВКТ-мен салыстырғанда 18,3%-ға төмендетеді. Абляция нәтижелері педагогикалық рефлексия ерекшеліктерінің РСК болжамына 9,2 F1 пунктін қосатынын, ТК және СК-ға шамалы пайда әкелетінін, із таксономиясының компонентке тән дизайнын растайтынын растайды. Ерекшеліктердің маңыздылығын талдау пернеаралық кідіріс орналастыру және компиляция қателерінің ауысу үлгілері СК бағалауы үшін ең ақпараттық сигналдар екенін, ал лексикалық РСК ерекшеліктері педагогикалық компонентті болжауда басым екенін көрсетеді.

Негізгі сөздер: білімді бақылау, градиентті күшейту, контейнерлік телеметрия, ТРАСК, мінез-құлық ізін талдау, бағдарламалау білімі, көп шығысты жіктеу.

МНОГОВЫХОДНОЙ ГРАДИЕНТНЫЙ БУСТИНГ ДЛЯ ОЦЕНКИ КОМПОНЕНТОВ ТРАСК НА ОСНОВЕ ТРАССИРОВКИ ПРОГРАММИРОВАНИЯ НА УРОВНЕ КОНТЕЙНЕРА**Аннотация**

Оценка одновременного развития компетенций в программировании и педагогических знаний предметной области у будущих учителей информатики представляет собой нерешенную проблему оценки. Инструменты, основанные на опросах, отражают убеждения и самооценки, а не фактическое поведение, а стандартные конвейеры анализа программирования ориентированы на оценку одного навыка, не моделируя измерение педагогических знаний, которое отличает кандидатов в учителя от обычных студентов, изучающих информатику. В данной статье представлена многовыходная архитектура градиентного бустинга, которая оценивает все четыре уровня освоения компонентов ТРАСК: технологические знания, педагогические знания, знания предметной области и педагогические знания предметной области, на основе поведенческих трасс, собранных на уровне среды выполнения контейнеров во время лабораторных занятий по программированию. Шестикатегориальная таксономия признаков выводится из потоков событий демона Docker, журналов компиляции на уровне приложений, последовательностей нажатий клавиш, снимков системы контроля версий, взаимодействий при запросе помощи и тонко настроенного классификатора педагогического отражения DistilBERT. Эксперименты на синтетическом эталонном наборе данных, включающем 850 аннотированных сессий, позволяют оценить восемь конфигураций модели в сравнении с четырьмя базовыми моделями, включая стандартную байесовскую трассировку знаний и градиентный бустинг с одним выходом. Предложенная архитектура достигает макроусредненного значения F1, равного 0,791, по всем четырем компонентам ТРАСК, превосходя самый сильный базовый показатель с одним выходом на 6,8 процентных пункта и снижая среднюю абсолютную ошибку при оценке непрерывного уровня знаний на 18,3% по сравнению с ВКТ. Результаты анализа подтверждают, что педагогические характеристики отражения вносят вклад в 9,2 пункта F1 в прогнозирование РСК, при этом для ТК и СК прирост был незначительным, что подтверждает компонентно-специфическую структуру таксономии трассировки. Анализ важности признаков показывает, что размещение пауз между клавишами и шаблоны перехода ошибок компиляции являются наиболее информативными сигналами для оценки СК, в то время как лексические характеристики РСК доминируют в прогнозировании педагогических компонентов.

Ключевые слова: трассировка знаний, градиентный бустинг, контейнерная телеметрия, ТРАСК, анализ поведенческих трасс, обучение программированию, многовыходная классификация.

REFERENCES

- 1 Saeli M., Perrenet J., Jochems W.M.G., Zwaneveld B. (2011) Teaching Programming in Secondary School: A Pedagogical Content Knowledge Perspective // Informatics in Education. Vol. 10. No. 1. P. 73—88 [in English]
- 2 Schulte C., Bell T., Settle A., (2011) Wörner K. Secondary School Informatics // Proceedings of the 16th Annual Conference Reports on Innovation and Technology in Computer Science Education (ITiCSE). P. 65—80. Available at – URL: <https://portal.fis.tum.de/en/publications/computer-scienceinformatics-in-secondary-education/> [in English] (accessed: 12.01.2026)
- 3 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. Vol. 29. No. 4. P. 775—806. Available at – URL: <https://link.springer.com/article/10.1007/s11257-019-09234-7> [in English] (accessed: 12.01.2026)
- 4 Robles-Gómez A., Tobarra L., Hernández R., Caminero A.C., Ros S., Pastor R. (2019) Analyzing the Students' Learning Within a Container-Based Virtual Laboratory for Cybersecurity // Proceedings of TEEM. [doi: 10.1145/3362789.3362819](https://doi.org/10.1145/3362789.3362819). [in English]
- 5 Hundhausen C.D., Agrawal A., Agarwal P. (2013) Talking About Code: Integrating Pedagogical Code Reviews into Early Computing Education // ACM Transactions on Computing Education. Vol. 13. No. 3. P. 1-28. Available at – URL: <https://dl.acm.org/doi/10.1145/2499947.2499951> [in English] (accessed: 23.01.2026)
- 6 Corbett A.T., Anderson J.R. (1994) Knowledge Tracing: Modeling the Acquisition of Procedural Knowledge // User Modeling and User-Adapted Interaction. Vol. 4. No. 4. P. 253-278. [doi: 10.1007/BF01099821](https://doi.org/10.1007/BF01099821). [in English]
- 7 Piech C., Spencer J., Huang J., Ganguli S., Sahami M., Guibas L., Ng A. (2015) Deep Knowledge Tracing // Advances in Neural Information Processing Systems (NeurIPS). Vol. 28. P. 505-513. [in English]
- 8 Ihantola P., Vihavainen A., Ahadi A., Butler M., Börstler 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. P. 41—63. [doi: 10.1145/2858796.2858798](https://doi.org/10.1145/2858796.2858798). [in English]
- 9 Leinonen J., Longi K., Klami A., Vihavainen A. (2016) Automatic Inference of Programming Performance from Typing Patterns // Proceedings of the 47th ACM Technical Symposium on Computing Science Education. P. 132—137. [doi: 10.1145/2839509.2844612](https://doi.org/10.1145/2839509.2844612). [in English]

- 10 Liu Z., Zhu C., Gasevic D. (2022) Educational Process Mining for Discovering Students' Problem-Solving Strategies in Programming Education // IEEE Transactions on Learning Technologies. Vol. 15. No. 3. P. 383—397. [doi: 10.1109/TLT.2022.3166228](https://doi.org/10.1109/TLT.2022.3166228). [in English]
- 11 Elmoazen R., Saqr M., Misiejuk M., Hirsto L. (2023) Learning Analytics in Virtual Laboratories: A Systematic Literature Review of Empirical Research // Smart Learning Environments. Vol. 10. No. 1. P. 1—23. [doi: 10.1186/s40561-023-00244-y](https://doi.org/10.1186/s40561-023-00244-y). [in English]
- 12 Khajah M., Lindsey R.V., Mozer M.C. (2016) How Deep Is Knowledge Tracing? // Proceedings of the 9th International Conference on Educational Data Mining (EDM). P. 94—101. [in English]
- 13 Barnes T. (2005) The Q-Matrix Method: Mining Student Response Data for Knowledge // Proceedings of the AAAI Workshop on Educational Data Mining. P. 39—46. [in English]
- 14 Baker R.S., Siemens G. (2014) Educational Data Mining and Learning Analytics // Cambridge Handbook of the Learning Sciences, 2nd ed. New York: Cambridge University Press. P. 253—274. Available at – URL: <https://doi.org/10.1017/CBO9781139519526.016> [in English] (accessed: 23.01.2026)
- 15 Lundberg S.M., Lee S.-I. (2017) A Unified Approach to Interpreting Model Predictions // Advances in Neural Information Processing Systems (NeurIPS). Vol. 30. [in English]
- 16 Hermans F., Aivaloglou E. (2017) Code Smells in Programming Education // Proceedings of the 2017 ACM Conference on Innovation and Technology in Computer Science Education (ITiCSE). P. 220—225. [doi: 10.1145/3059009.3059061](https://doi.org/10.1145/3059009.3059061). [in English]
- 17 Denny P., Luxton-Reilly A., Tempero E., Hendrickx J. (2011) Understanding the Syntax Errors of Novices // Proceedings of the 16th Annual Joint Conference on Innovation and Technology in Computer Science Education (ITiCSE). P. 213—217. [doi: 10.1145/1999747.1999807](https://doi.org/10.1145/1999747.1999807). [in English]
- 18 Sanh V., Debut L., Chaumond J., (2019) Wolf T. DistilBERT, a Distilled Version of BERT: Smaller, Faster, Cheaper and Lighter // arXiv preprint arXiv:1910.01108. [in English]
- 19 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. Vol. 42. No. 2. P. 123—149. [doi: 10.1080/15391523.2009.10782544](https://doi.org/10.1080/15391523.2009.10782544). [in English]
- 20 Qian Y., Lehman J. (2017) Students' Misconceptions and Other Difficulties in Introductory Programming // ACM Transactions on Computing Education. Vol. 18. No. 1. P. 1—24. [doi: 10.1145/3077618](https://doi.org/10.1145/3077618). [in English]
- 21 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. Vol. 26. No. 1. P. 205—223. [doi: 10.1007/s40593-015-0089-1](https://doi.org/10.1007/s40593-015-0089-1) [in English]
- 22 Mishra P., Koehler M.J. (2006) Technological Pedagogical Content Knowledge: A Framework for Teacher Knowledge // Teachers College Record. Vol. 108. No. 6. P. 1017-1054. [doi: 10.1111/j.1467-9620.2006.00684.x](https://doi.org/10.1111/j.1467-9620.2006.00684.x). [in English]
- 23 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. Vol. 31. No. 4. P. 897-912. [doi: 10.1002/cae.22622](https://doi.org/10.1002/cae.22622). [in English]
- 24 Malan D.J., Yu J., Kutaula A., Schmitt J. (2022) Teaching CS50 via Containers // Proceedings of the 27th ACM Conference on Innovation and Technology in Computer Science Education. P. 293—299. [doi: 10.1145/3502717.3532137](https://doi.org/10.1145/3502717.3532137). [in English]
- 25 Baker R.S., Hawn A. (2022) Algorithmic Bias in Education // International Journal of Artificial Intelligence in Education. Vol. 32. No. 4. P. 1052—1092. [doi: 10.1007/s40593-021-00285-9](https://doi.org/10.1007/s40593-021-00285-9). [in English]

Information about authors:

Duisenova Gaukhar - **corresponding author**, doctoral student, L.N. Gumilyov Eurasian National University, Astana, Republic of Kazakhstan

E-mail: gauhar.duisen9@gmail.com

ORCID: <https://orcid.org/0009-0006-0404-4299>

Shyndaliyev Nurzhan - candidate of pedagogical sciences, associate professor, L.N. Gumilyov Eurasian National University, Astana, Republic of Kazakhstan

E-mail: nurzhan-11@list.ru

ORCID: <https://orcid.org/0000-0001-5284-3526>

Shadiev Rustam - professor, PhD, Zhejiang University Hangzhou, China

E-mail: rustamsh@gmail.com

ORCID: <https://orcid.org/0000-0001-5571-1158>

Информация об авторе:

Дуйсенова Гаухар Асылхановна - **основной автор**, докторант, Евразийский национальный университет имени Л.Н. Гумилева, г. Астана, Республика Казахстан

E-mail: gauhar.duisen9@gmail.com

ORCID: <https://orcid.org/0009-0006-0404-4299>

Шындалиев Нуржан Тажибаевич - кандидат педагогических наук, ассоциированный профессор, Евразийский национальный университет имени Л.Н. Гумилева, г. Астана, Республика Казахстан

E-mail: nurzhan-11@list.ru

ORCID: <https://orcid.org/0000-0001-5284-3526>

Шадиёв Рустам Нарзикулович - PhD, профессор, Чжэцзянский университет, г. Ханчжоу, Китай

E-mail: rustamsh@gmail.com

ORCID: <https://orcid.org/0000-0001-5571-1158>

Авторлар туралы мәлімет:

Дуйсенова Гаухар Асылхановна - **негізгі автор**, докторант, Л.Н.Гумилев атындағы Еуразия ұлттық университетінің, Астана қ., Қазақстан Республикасы

E-mail: gauhar.duisen9@gmail.com

ORCID: <https://orcid.org/0009-0006-0404-4299>

Шындалиев Нуржан Тажибаевич – қауымдастырылған профессор, доцент, педагогика ғылымдарының кандидаты, Л. Н. Гумилев атындағы Еуразия ұлттық университеті, Астана қ., Қазақстан Республикасы

E-mail: nurzhan-11@list.ru

ORCID: <https://orcid.org/0000-0001-5284-3526>

Шадиёв Рустам Нарзикулович - PhD, профессор, Чжэцзян университеті, Ханчжоу қ., Қытай

E-mail: rustamsh@gmail.com

ORCID: <https://orcid.org/0000-0001-5571-1158>