

Frontier AI Model Output for Educational Application Development via Vibe-Coding: A Systematic Literature Review Comparing Western and Eastern AI Models

¹Muhammad Haziq Abd Rashid*, ²Hairul Azhar Mohamad, ³Pavithran Ravinthra Nath, ⁴Amir Lukman Abd Rahman, ⁵Muhammad Luthfi Mohaini, ⁶Muhammad Hazim Abd Rashid, ⁷Urai Salam

¹²³⁴⁵Akademi Pengajian Bahasa Universiti Teknologi MARA

⁶Malaysian Communications and Multimedia Commission

⁷Universitas Tanjungpura Kalimantan

*Corresponding Author

DOI: <https://doi.org/10.47772/IJRISS.2026.102100117>

Received: 07 September 2026; Accepted: 17 September 2026; Published: 01 October 2026

ABSTRACT

Contemporary artificial intelligence models can translate natural-language prompts into executable code, a practice that practitioners have termed vibe-coding. The capability is offered by Western developers (OpenAI, Anthropic, Google, Microsoft) and Eastern developers (DeepSeek, Alibaba, Zhipu AI, Moonshot AI) alike, yet no systematic comparison of their output quality in educational settings has been published. This review addresses that gap. Drawing on the Technological Pedagogical Content Knowledge framework and Rogers' Diffusion of Innovation theory, a PRISMA 2020 systematic review was conducted on 42 studies published between January 2022 and August 2026 and retrieved from Scopus, Web of Science, IEEE Xplore and the ACM Digital Library. Each study was assessed against six dimensions: code correctness, code quality, hallucination rates, natural language understanding, developer productivity and pedagogical appropriateness. The evidence indicates convergence rather than supremacy. Eastern models perform within the range established by Western models across several coding benchmarks, and independent testing places DeepSeek V3.2 inside the hallucination band occupied by four Western frontier models. The supporting evidence, however, remains markedly asymmetrical. Every one of the seven Eastern-model entries in the corpus is a technical report published by the organisation that built the model, and this holds for the 2026 frontier releases as firmly as for their predecessors. One 2026 flagship release was excluded outright because its technical report states no measurable capability outcome at all. Western models have been examined through classroom deployments, independent audits and large-sample quasi-experimental designs, whereas Eastern models are evaluated predominantly through technical reports issued by their own developers. Work published in 2026 has begun to address vibe-coding security and cross-model hallucination, but no head-to-head classroom comparison of Western and Eastern models has yet been reported. That absence is identified as the principal gap in the field. The review proposes a reusable six-dimension framework for future comparative research and sets out the implications for educators and policymakers operating within an increasingly bipolar artificial intelligence landscape in Southeast Asia and beyond.

Keywords: systematic literature review, vibe-coding, AI-assisted code generation, frontier AI models, code output quality

INTRODUCTION

The public release of ChatGPT in November 2022 marked the beginning of a period of rapid change in AI-assisted software development. GPT-4, Claude, Gemini and a range of open-weight alternatives followed, each generating

code with greater fluency than most practitioners had anticipated. By early 2025 the practice had acquired a name. Karpathy (2025) described a new mode of working in which the developer states an intention in ordinary language and the model produces the implementation, observing that the programmer may 'fully give in to the vibes, embrace exponentials, and forget that the code even exists'. The term was adopted rapidly across professional and academic discourse.

The implications extend well beyond commercial software development. Students now enter programming classrooms with tools capable of writing, explaining and debugging code on demand. The question facing computing educators is no longer whether AI-assisted development will reshape the teaching of programming, but how that reshaping will occur and which models will mediate it.

The second half of that question carries a geopolitical dimension, and educators in Southeast Asia encounter it directly. AI coding assistance has divided along broadly geographic lines, and Western models dominate international markets and the English-language research literature, among them the GPT family (OpenAI), Claude (Anthropic), Gemini (Google DeepMind) and GitHub Copilot. Eastern models present a different case. DeepSeek (DeepSeek AI), Qwen (Alibaba Cloud), GLM and CodeGeeX (Zhipu AI) and Kimi (Moonshot AI) have achieved comparable benchmark results and hold a large share of the market within China. They are nonetheless cited infrequently in the journals that international educators most often consult.

The consequences are practical. Consider a lecturer selecting an AI assistant for an introductory programming course at a Malaysian or Indonesian institution, where the decision extends well beyond tool preference. It reaches the training data from which students will learn, the programming languages the model treats as defaults, the cost of sustained classroom use and the regulatory regime governing the data. Where only Western models have been examined in classroom research, the evidence informing such a decision is incomplete.

Several reviews have mapped the field of AI code generation (Umama et al., 2025; Raihan et al., 2025) and its educational consequences (R. Deng et al., 2024; Becker et al., 2023). None was framed around the Western and Eastern division, and none used that framing to compare output quality across the dimensions relevant to educational application development. That gap is addressed here. Two theoretical lenses are applied, the Technological Pedagogical Content Knowledge framework (Mishra & Koehler, 2006) and Rogers' (2003) Diffusion of Innovation theory. Four research questions are posed.

RQ1. What is the current state of research on frontier AI models, Western and Eastern alike, for AI-assisted code generation in educational application development contexts between 2022 and 2026?

RQ2. How do the code outputs of Western-origin AI models (GPT, Claude, Gemini, Copilot) compare with those of Eastern-origin AI models (DeepSeek, Qwen, GLM, Kimi and others) on code correctness, code quality, hallucination rates and natural language understanding?

RQ3. What impacts on developer productivity and pedagogical appropriateness are reported when these models are used for vibe-coding in educational or education-adjacent settings?

RQ4. What gaps, limitations and future research directions emerge from the existing literature on geographically distinct AI coding assistants for education?

The remainder of the paper is organised as follows. Section 2 establishes the theoretical framework and reviews the relevant literature, Section 3 describes the PRISMA 2020 methodology, and Section 4 presents the results. Section 5 discusses the findings and Section 6 concludes.

LITERATURE REVIEW

Theoretical Framework

Two theoretical perspectives anchor this review, and each performs a distinct function. The Technological Pedagogical Content Knowledge framework identifies what should be measured when AI coding models are evaluated for educational use, while Diffusion of Innovation theory explains why those measurements are missing for so many of the models available.

Mishra and Koehler (2006) developed the Technological Pedagogical Content Knowledge framework around three overlapping knowledge domains: technological, pedagogical and content knowledge. Effective integration of educational technology occurs where all three intersect, and the technology there serves both the subject matter and the teaching approach. In the present context the AI model constitutes the technology, programming concepts and computational thinking constitute the content, and instructional strategy, assessment design and scaffolding constitute the pedagogy. Evaluating the technology in isolation from the other two domains leaves the account incomplete.

The six comparison dimensions used here map onto that structure, with D1 to D4 falling within technological knowledge, since code correctness, code quality, hallucination rates and natural language understanding all measure the capability of the tool in isolation from any teaching context. D5, developer productivity, spans the technological and pedagogical boundary, since it is the point where model capability meets classroom workflow and instructor time. D6, pedagogical appropriateness, sits at the intersection itself, and there the tool must serve the learning objectives and the instructional approach at once. A model that generates correct code but cannot articulate its reasoning may score well on D1 while failing D6 entirely. The distinction is easily missed. Most benchmark studies evaluate D1 alone, addressing roughly one third of what matters in the terms of this framework.

Rogers (2003) proposed that innovations diffuse according to five attributes: relative advantage, compatibility, complexity, trialability and observability. Applied to the Western and Eastern division, these attributes make the asymmetry in the literature predictable rather than anomalous. Entering the market earlier gave Western models relative advantage through timing alone. They suited English-language research conventions, their documented interfaces supported straightforward trialability, and extensive media coverage produced high observability. Eastern models face barriers on every one of these attributes despite benchmark performance that now approaches or matches their Western counterparts. Export controls, linguistic friction beyond Chinese-speaking markets and limited visibility in English-language journals are all implicated.

One consequence of this pattern warrants its own name, and it is termed pedagogical model lock-in here and treated as Theme T6 in Section 4.7, a phenomenon that is diffusional rather than evaluative. Educational platforms built on Western models generate path dependency, though not because GPT or Claude are demonstrably better suited to teaching; their earlier adoption simply produced the evidence, the tooling and the familiarity that make them the least resistant option. The loop then reinforces itself. Tested tools are adopted by educators, adopted tools are then tested by researchers, and alternatives are left with no evidence at all. Breaking that loop requires precisely the cross-bloc comparative research this review finds to be absent. The Technological Pedagogical Content Knowledge framework specifies the right questions, and Diffusion of Innovation theory explains why so many of the answers are missing.

Vibe-Coding and AI-Assisted Code Generation

Generating code from natural language is not a recent proposition. Earlier work on program synthesis and intent-based programming explored comparable territory, although practical capability remained limited until OpenAI released Codex in 2021 and integrated it into GitHub Copilot. Those tools altered the nature of the development task, so that rather than composing code statement by statement, the programmer now specifies the intended outcome, reviews what the model returns and refines it through conversational prompts.

Karpathy's (2025) formulation condensed this workflow into a single expression. Adoption was rapid, because the term named an experience many developers recognised but had not previously labelled. That experience is the condition in which tooling friction falls low enough for a developer to work at the level of intent rather than syntax.

Academic investigation of the practice is recent but expanding quickly, and Pimenova et al. (2025) conducted the first systematic qualitative study, drawing on more than 190,000 words from interviews, Reddit discussions and LinkedIn posts to map developer perceptions of trust, co-creation and flow. They characterise vibe-coding as a negotiation with the model, one demanding metacognitive skills such as prompt formulation and iterative refinement. Huang et al. (2025) reported a contrasting picture. Their professional participants rejected the vibe label altogether and described their use of AI as controlled and deliberate. Subsequent work has moved towards direct measurement of outcomes (Uddin, 2026) and towards how prior developer experience shapes practice (Fawzy et al., 2026), extending the evidence base beyond its initial qualitative phase.

A growing body of benchmark-oriented research underlies these qualitative accounts. HumanEval, MBPP and SWE-bench have become the standard instruments for measuring whether a model produces correct code. Each has been criticised for saturation and contamination. Reviewing 58 studies under PRISMA guidance, Umama et al. (2025) found that HumanEval and MBPP remain dominant but are approaching ceiling effects for frontier models. Yetiştiren et al. (2023) extended the scope of measurement beyond correctness to validity, security, reliability and maintainability, reporting wide variation across GitHub Copilot, Amazon CodeWhisperer and ChatGPT even on identical HumanEval prompts.

Frontier AI Models: The Western and Eastern Landscape

The frontier AI model ecosystem can be partitioned geographically, and the Western group includes the GPT family (OpenAI), Claude (Anthropic), Gemini (Google DeepMind) and GitHub Copilot (Microsoft and GitHub), all of which have benefited from large-scale investment, extensive English-language training data and early market entry. Eastern models, concentrated in China, include DeepSeek (DeepSeek AI), Qwen (Alibaba Cloud), GLM and CodeGeeX (Zhipu AI) and Kimi (Moonshot AI), alongside offerings from ByteDance (Doubao), Baidu (ERNIE Bot) and several smaller laboratories.

The performance distance between the two groups has narrowed sharply, though the picture is more complex than benchmark headlines suggest. On HumanEval, DeepSeek-V3 records a pass@1 score of 65.2, marginally below the 67.0 reported for GPT-4 (DeepSeek AI, 2024b). DeepSeek-Coder-V2 is built on a mixture-of-experts architecture and reports performance superior to GPT-4 Turbo on coding and mathematical benchmarks, with programming language support expanded from 86 to 338 (DeepSeek AI, 2024a). The Qwen2.5-Coder series reports state-of-the-art results across more than ten benchmarks spanning code generation, completion, reasoning and repair (Hui et al., 2024). Taken together, these results indicate that Eastern models operate within the range established by Western models. That range is not uniformly surpassed.

Releases from 2026 continue this pattern at greater scale. GLM-5, a 744-billion-parameter model whose technical report is titled with the transition from vibe coding to agentic engineering, reports state-of-the-art results on open coding benchmarks (GLM-5 Team, 2026). Qwen3-Coder-Next, an 80-billion-parameter model activating 3 billion parameters at inference, is specialised for coding agents and evaluated on SWE-Bench and Terminal-Bench (Cao et al., 2026). Kimi K3 is a 2.8-trillion-parameter mixture-of-experts model with 104 billion activated parameters reporting frontier-level performance on long-horizon coding (Kimi Team, 2026). A fourth release, DeepSeek-V4, was identified by the search but did not meet the inclusion criteria: its preview technical report documents architecture and inference efficiency without reporting a measurable outcome on any comparison dimension. The exclusion is recorded in the Appendix and returned to in Section 5.4.

These figures warrant careful interpretation for a further reason, since almost every evaluation of an Eastern model originates with the organisation that built it. DeepSeek AI published both the DeepSeek-V3 and DeepSeek-Coder-

V2 technical reports, and the Qwen team published the Qwen2.5-Coder and Qwen3 reports. Independent third-party evaluation on a scale comparable to the external scrutiny applied to Western models remains sparse. Self-evaluation is not confined to Eastern developers, since OpenAI and Anthropic publish their own benchmark results too. The imbalance in outside scrutiny is nevertheless sharper. In Rogers' (2003) terms, that disparity reduces the observability of Eastern model performance within the international research community.

AI in Educational Software Development

AI code generation tools have entered educational settings rapidly, though unevenly. Kazemitabaar et al. (2024) deployed CodeAid, an assistant built on GPT-3.5, across a 700-student introductory programming course over twelve weeks, designing it to withhold direct code solutions and offer pseudocode and conceptual explanations instead. Thematic analysis of 8,000 student interactions reported 79% response accuracy and 86% perceived helpfulness. Whether the independent coding ability of students improved was not tracked by the study. Within the Technological Pedagogical Content Knowledge framework, CodeAid represents a deliberate attempt to optimise the technological and pedagogical interaction by constraining model output for teaching purposes.

Sun et al. (2024) addressed the learning question more directly, identifying in a quasi-experimental study of college students using ChatGPT-facilitated programming that effects on programming behaviours and performance varied in direction by task type. Fan et al. (2025) conducted a two-year comparison across 2023 and 2024 with 234 undergraduates in a Java web development course, contrasting AI-assisted pair programming with traditional human pairs and with individual work. Significantly higher intrinsic motivation and lower programming anxiety were produced by the AI-assisted condition than by individual programming, with outcomes comparable to human-human pairing. The collaborative depth and social presence of human-human pairing were not fully reproduced.

Larger-scale studies have appeared more recently. Huesca et al. (2026) tested four generative AI-supported instructional strategies in an introductory programming course involving 686 students across 53 class groups drawn from ten campuses and taught by 32 professors, providing the most statistically powered evidence for AI-assisted programming pedagogy currently available. Guo et al. (2026) examined the effect on the computational thinking of 83 tenth-grade students through a mixed-methods design.

A limitation is common to all of these studies: each evaluates Western models exclusively. CodeAid operates on GPT-3.5, and the pair programming study used a ChatGPT-based assistant. The systematic review and meta-analysis conducted by R. Deng et al. (2024), which examined 69 experimental studies of the effect of ChatGPT on student learning, likewise addresses a single family of Western models. No equivalent body of research exists for DeepSeek, Qwen or any other Eastern model in an educational context. The diffusion framework proposed by Rogers (2003) predicts this outcome: the models with the greatest observability and compatibility within English-language educational research are those that are studied, and the pattern reinforces itself.

Research Gap

The literature in its current state exhibits a structural imbalance. Research on code output quality is advancing rapidly, and work on AI in programming education continues to grow. These two streams have not, however, been brought together under a comparative framework that accounts for where the models under evaluation originate. Assessment of Western models proceeds both technically and pedagogically, whereas Eastern models are assessed almost exclusively through self-published technical reports, and their performance in educational settings remains unexamined. The Technological Pedagogical Content Knowledge framework identifies the cost: benchmark-only evaluations capture technological knowledge and leave the pedagogical and content dimensions untested.

This review is positioned within that gap. It synthesises evidence from both research traditions under a unified six-dimension framework grounded in the Technological Pedagogical Content Knowledge framework and in Diffusion of Innovation theory, then identifies the areas where comparative work is most acutely needed.

METHODOLOGY

Research Design

This review follows the PRISMA 2020 guidelines for systematic reviews (Page et al., 2021). The review protocol was developed a priori and covers database selection, search string construction, inclusion and exclusion criteria, quality appraisal, data extraction and thematic analysis. The review was not registered in a public registry.

Search Strategy

Four electronic databases were searched: Scopus, Web of Science, IEEE Xplore and the ACM Digital Library. Google Scholar served a supplementary role for backward and forward citation chasing. All searches were conducted in August 2026 and covered literature published from January 2022 to August 2026. No eligible study predating 2023 was identified, which is itself indicative of how recently the field has formed.

The search strategy combined three concept groups with the AND operator. Within each group, synonyms were joined by OR.

Group A (AI-assisted coding and vibe-coding): “vibe coding” OR “vibe-coding” OR “AI-assisted code generation” OR “LLM-based coding” OR “AI pair programming” OR “natural language to code” OR “prompt-driven development” OR “AI code generation” OR “large language model coding”.

Group B (educational context): “educational app” OR “learning application” OR “education* software” OR “e-learning development” OR “coding education” OR “CS education” OR “programming education” OR “educational technology” OR “computer science education”.

Group C (AI models): “GPT” OR “ChatGPT” OR “Copilot” OR “GitHub Copilot” OR “Claude” OR “Gemini” OR “Codex” OR “DeepSeek” OR “Qwen” OR “GLM” OR “ChatGLM” OR “CodeGeeX” OR “Kimi” OR “Doubao” OR “ERNIE” OR “MiniMax” OR “frontier model” OR “large language model” OR “code LLM” OR “AI model comparison”.

The combined query was Group A AND Group B AND Group C. Because the education-specific literature proved thin, yielding fewer than 15 studies, the search was broadened in accordance with the pre-registered thin-literature strategy to include studies in general software development contexts that reported on at least one of the six comparison dimensions. This broadening is reported transparently and is treated as a finding in its own right.

An update search was conducted on 1 September 2026. Its purpose was to capture records published towards the end of the window and to verify how complete the initial retrieval had been. It re-executed Group A AND Group C without the educational restriction, on the reasoning that model technical reports rarely carry educational descriptors, and added forward citation chasing from the 2026 records already retrieved. The update returned nine additional in-window records that the initial search had not surfaced. Seven met the inclusion criteria and passed quality appraisal. One was excluded at eligibility for reporting no measurable outcome on any comparison dimension, and one at appraisal for falling below the quality threshold; both are listed in the Appendix. The initial search had therefore under-retrieved. The shortfall is documented rather than concealed. The clearest instance is a model technical report whose title contains the phrase vibe coding, which the Group A string should have matched. This is reported as a limitation of the original search rather than as a change of scope.

Inclusion and Exclusion Criteria

Table 1 presents the inclusion criteria. The exclusion criteria applied during screening and eligibility assessment are given in Table 2.

The eligibility rule requires comment, since it departs from the convention that a systematic review admits peer-reviewed work alone. Three considerations governed the decision. The field moves faster than journal publication cycles, so a rule restricted to peer-reviewed outputs would have excluded most evidence published after 2025. Model capability is reported first, and frequently only, in developer technical reports, so excluding that class would have removed every Eastern model from the review and with it the comparison the review exists to make. Peer review is therefore replaced rather than waived. Every record, whatever its class, was required to pass the appraisal instrument in Table 3 at a threshold of 4.0 out of 8.0, and the cost of that substitution is recorded in the scores themselves: vendor technical reports cluster at the bottom of the retained range and return zero on threats to validity, as Section 4.1 reports. Commercial benchmark aggregators were excluded outright under E5, and no claim in this review rests upon them.

Table 1 Inclusion criteria

Code	Inclusion Criteria
I1	Published between January 2022 and August 2026
I2	Peer-reviewed journal articles, conference papers or workshop papers; preprints reporting a measurable outcome on at least one comparison dimension; or model technical reports issued by model developers. All three classes are admitted only on passing quality appraisal (Table 3, threshold 4.0 of 8.0)
I3	Empirically evaluates or compares AI model output for code generation
I4	Involves educational or education-adjacent application development context (broadened to include general software development per thin-literature strategy)
I5	Written in English or Mandarin Chinese (with English abstract)
I6	Reports measurable outcomes on at least one of six comparison dimensions (D1 to D6)

Table 2 Exclusion criteria

Code	Exclusion Criteria
E1	Opinion pieces, editorials, or magazine articles without empirical data
E2	Studies exclusively on AI for non-coding tasks (text generation, image synthesis)
E3	Duplicate publications (the most complete version was retained)
E4	Secondary sources used as primary evidence, that is, reviews or meta-commentaries cited for outcomes they did not themselves generate. Secondary sources are retained where they serve a corroborating or contextual function
E5	Non-appraisable grey literature: blog posts, marketing material, commercial benchmark aggregators, and preprints or technical reports falling below the quality appraisal threshold
E6	Studies evaluating models exclusively older than GPT-3.5 (pre-frontier era)

Study Selection Process

Figure 1 presents the PRISMA flow. The initial search across four databases returned 220 records, and after duplicates were removed, 112 unique records proceeded to title and abstract screening. At that stage, 60 records were excluded as opinion or grey literature (E1 and E5), work on non-coding AI (E2), studies without original data (E4), studies using pre-frontier models exclusively (E6), or publications outside the defined timeframe (I1). The remaining 52 records underwent full-text review and quality appraisal, during which 17 were excluded for insufficient empirical data, unverifiable references or duplicate content. A corpus of 35 studies was retained at that point. The update search described in Section 3.2 contributed a further nine records, all of which proceeded to full-text review; one was excluded at eligibility and one at appraisal, so seven were retained. The final corpus comprised 42 studies.

Two reviewers screened titles, abstracts and full texts independently, resolving disagreements through discussion. A formal inter-rater agreement coefficient was not computed.

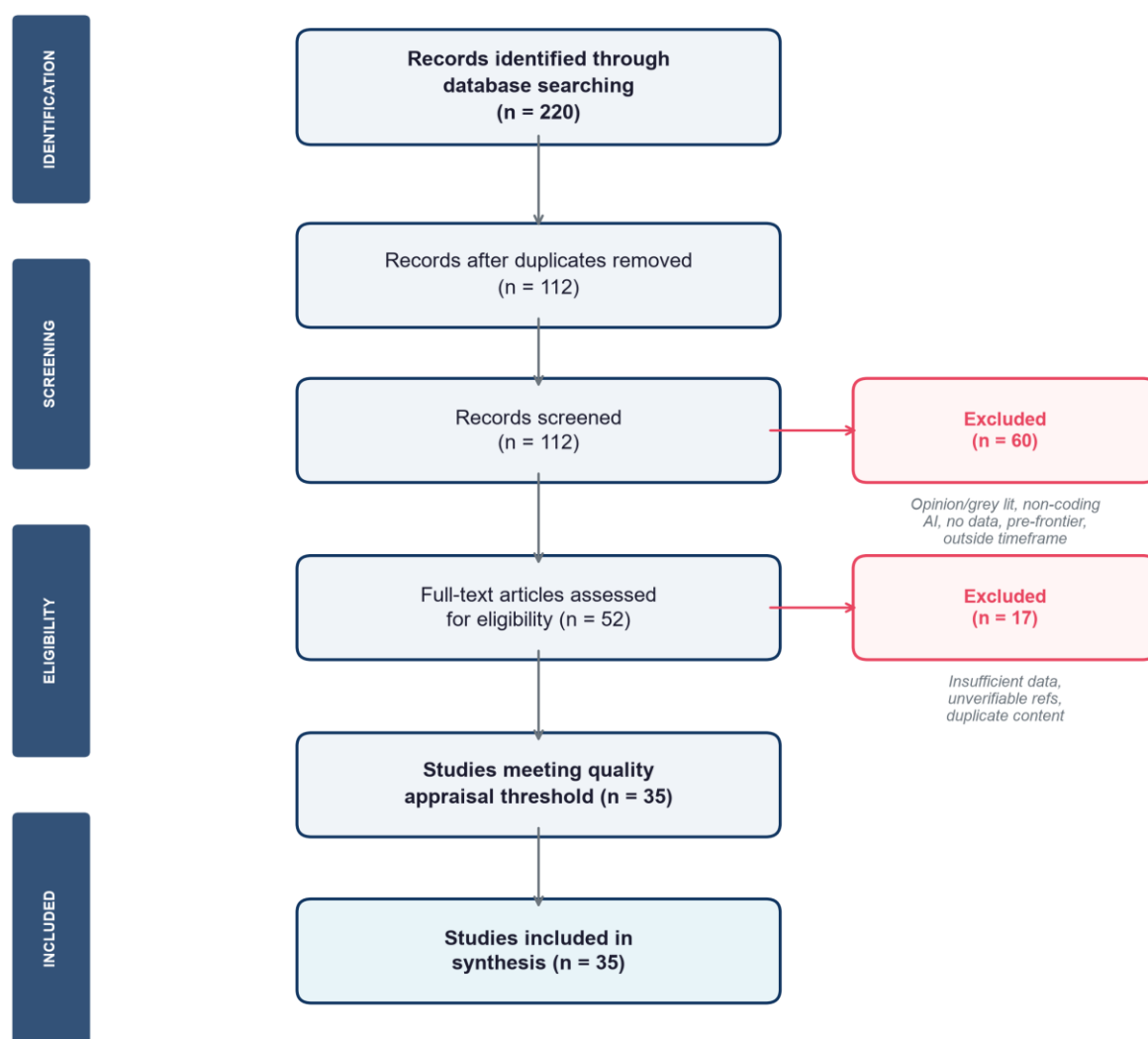


Figure 1 PRISMA 2020 flow diagram of the study selection process

Quality Appraisal

Each study was appraised against eight criteria, scored 0, 0.5 or 1, to a maximum of 8.0. Table 3 presents the appraisal instrument.

Table 3 Quality appraisal criteria

Code	Criterion	Score
QA1	Research objectives clearly stated	0 / 0.5 / 1
QA2	AI model(s) and version clearly identified	0 / 0.5 / 1
QA3	Evaluation methodology described with sufficient replication detail	0 / 0.5 / 1
QA4	Benchmark/dataset and evaluation metrics explicitly defined	0 / 0.5 / 1
QA5	Sample size or evaluation scope adequate	0 / 0.5 / 1
QA6	Threats to validity or limitations discussed	0 / 0.5 / 1
QA7	Findings supported by reported data	0 / 0.5 / 1
QA8	Study context (educational/development setting) clearly described	0 / 0.5 / 1

Studies scoring below 4.0 out of 8.0 were excluded from the synthesis and are listed in the Appendix. Of the 43 studies that reached appraisal, 42 met the threshold and one did not. Scores across the retained corpus ranged from 5.0 to 7.5. Model technical reports occupy the lower bound, scoring consistently on objectives, model identification and evaluation scope, yet returning zero on threats to validity, since a section of that kind is not conventionally carried in a vendor report.

Data Extraction and Synthesis

Data were extracted along four axes. The first was bibliometric: authors, year, venue, country of affiliation and study type. The second was technical, covering models evaluated, model origin, task domain, benchmarks and programming languages. The third recorded outcomes across the six comparison dimensions (D1, code correctness and functionality; D2, code quality and maintainability; D3, hallucination and fabrication; D4, natural language understanding; D5, developer productivity; D6, pedagogical appropriateness). The fourth captured educational context, including level, subject area, pedagogical approach and application type. The six dimensions were selected to span the full Technological Pedagogical Content Knowledge intersection. D1 to D4 assess technological knowledge, D5 bridges technological and pedagogical knowledge, and D6 addresses the intersection itself.

A fifth axis records evidence independence. Each study was assigned a tier according to the relationship between the evaluator and the system evaluated. Tier E-I covers independent empirical and classroom studies, in which researchers unconnected to the model developer observe the model in use with human participants. Tier E-II covers independent benchmarks and audits, in which unconnected researchers measure model output under controlled conditions without human participants. Tier E-III covers developer self-reports, in which the organisation that built the model reports its own evaluation of it. The tier is recorded for every study in Appendix Table A1 and carried into the comparison matrix in Table 6, so that evidence differing in independence is not read as equivalent.

Because the six dimensions are offered for reuse beyond this review, each is defined operationally in Table 4, which states the construct, the evidence admitted against it, the metrics conventionally used to report it, and what is deliberately excluded from it. The exclusions carry as much weight as the inclusions: D1 and D2 are separated by an execution criterion, D3 is confined to entities that can be checked against a registry, D4 distinguishes comprehension from coverage, D5 requires a comparison condition, and D6 requires a learner.

Table 4 Operational definitions and measurement criteria for the six comparison dimensions

Dimension	Construct	Admissible evidence	Typical metrics	Not counted under this dimension
D1 Code correctness and functionality	Whether generated code executes and produces the behaviour the prompt specifies.	Executable evaluation against a defined task set with a pass or fail criterion, by automated test suite, online judge, or human verification of the functional outcome.	pass@k; compilation success rate; test-case pass rate; accepted-solution rate; task success rate.	Capability claims carrying no benchmark, dataset or metric, which criterion I6 excludes; judgements of code quality made without an execution criterion, which belong to D2.
D2 Code quality and maintainability	Properties of generated code that persist once it runs correctly: structure, readability, security and maintenance cost.	Static analysis, security audit, or rubric-scored human assessment applied to the generated artefact, with the instrument named.	technical debt minutes per function; vulnerability counts by CWE category; maintainability or readability scores; proportion of outputs judged secure.	Functional pass rates, which belong to D1; developer perception of quality unaccompanied by assessment of the artefact itself.
D3 Hallucination and fabrication	Generation of references to software entities that do not exist.	Systematic validation of generated identifiers against an authoritative registry or API specification, with the reference list named and the sample size reported.	hallucinated package rate; proportion of unique fabricated names; persistence of a fabrication across repeated prompts.	Factual error in natural-language explanation; incorrect use of an API that does exist, which belongs to D1. The corpus measures package hallucination only, so behavioural and API-level hallucination remain unmeasured, as Section 4.4.3 records.
D4 Natural language understanding	The capacity to recover the intent a prompt expresses, including prompts not written in English.	Comparison of outcomes across prompt languages, phrasings or ambiguity conditions, holding the task constant.	success rate by prompt language; token consumption by prompt language; resolution rate on ambiguous specifications.	Counts of supported programming or natural languages, which state coverage rather than comprehension, as Section 4.4.4 sets out.

D5 Developer productivity	Change in the rate or effort of task completion when the model is used, measured at the level of the workflow rather than the artefact.	Comparison against a control or baseline condition, with the unit of work defined.	tasks completed per unit time; time to completion; iteration or attempt count; self-reported helpfulness where paired with an objective measure.	Benchmark scores standing alone; perceived usefulness reported without a comparison condition.
D6 Pedagogical appropriateness	Whether the model serves the learning objective rather than only the coding task. Requires a learner-facing outcome.	Study conducted with learners in an instructional setting, reporting an outcome referred to learning, affect, help-seeking behaviour or assessment performance.	learning gain; computational thinking measures; motivation or anxiety scales; help-seeking judgements of explanation quality.	Any evaluation conducted without learner participants. This exclusion is the reason no Eastern model qualifies on D6, as Section 4.5.2 reports.

Synthesis proceeded in two layers. The first was descriptive and quantitative. It comprised frequency tables, a comparison matrix mapping models to dimensions, and a geographic analysis of author affiliations. The second applied the six-phase reflexive thematic analysis described by Braun and Clarke (2006): familiarisation, initial coding, searching for themes, reviewing themes, defining and naming themes, and producing the report. Six themes were hypothesised a priori. Each was then tested against the data. The corpus does not support meta-analysis, since no two studies share a design, an outcome measure and a comparator, and the developer technical reports supply no variance estimates. The synthesis is therefore a structured narrative synthesis, conducted within study design and independence tiers and reported in accordance with the Synthesis Without Meta-analysis guideline (Campbell et al., 2020).

RESULTS

Study Selection

The PRISMA process described in Section 3.4 yielded 42 studies that met the quality appraisal threshold. The mean quality score of 6.5 out of 8.0 indicates a generally well-conducted evidence base, and scores clustered by study type. Empirical and quasi-experimental studies scored between 6.0 and 7.5, typically losing marks where an educational context descriptor was absent (QA8 = 0) or model versions were imprecise (QA2 = 0.5). All seven model technical reports scored 5.0, losing marks uniformly on threats to validity (QA6 = 0) and partially on replication detail (QA3 = 0.5) and benchmark definition (QA4 = 0.5). That the two study types separate so cleanly on the instrument is itself informative. The point is taken up again in Section 4.3.

Descriptive Overview

Table 5 lists the 42 included studies with their bibliometric details, the models tested, model origin and study type.

Table 5 Overview of included studies

ID	Authors	Year	Venue	Models Tested	Origin	Type	Independence
S01	Almanasra et al.	2026	Array	ChatGPT-4o, DeepSeek-R1	Both	Comparative	E-II
S02	Manik	2025	arXiv	ChatGPT-o1, DeepSeek-R1	Both	Comparative	E-II
S03	Sobo et al.	2024	Applied Artificial Intelligence	ChatGPT 3.5, Gemini 1.5 Pro, Claude 3.5 Sonnet	Western	Comparative	E-II
S04	Zhu et al.	2024	arXiv	12 models incl. GPT-4, Claude, DeepSeek-Coder, Qwen	Both	Benchmark	E-II
S05	Kazemitabaar et al.	2024	CHI 2024	GPT-3.5 (CodeAid)	Western	Classroom	E-I
S06	Sun et al.	2024	Int. J. Educ. Technol. High. Educ.	ChatGPT	Western	Quasi-experimental	E-I
S07	Wermelinger	2023	SIGCSE 2023	GitHub Copilot	Western	Empirical	E-II
S08	Fan et al.	2025	Int. J. STEM Educ.	GPT-3.5 Turbo, Claude 3 Opus	Western	Quasi-experimental	E-I
S09	Ziegler et al.	2024	Communications of the ACM	GitHub Copilot	Western	Field experiment	E-I
S10	Liao et al.	2024	IEEE Trans. Learn. Technol.	ChatGPT	Western	Quasi-experimental	E-I
S11	DeepSeek AI	2024a	arXiv	DeepSeek-Coder-V2	Eastern	Technical report	E-III
S12	Hui et al.	2024	arXiv	Qwen2.5-Coder	Eastern	Technical report	E-III
S13	DeepSeek AI	2024b	arXiv	DeepSeek-V3	Eastern	Technical report	E-III
S14	Qwen Team	2025	arXiv	Qwen3	Eastern	Technical report	E-III
S15	Yetiştiren et al.	2023	arXiv	Copilot, CodeWhisperer, ChatGPT	Western	Comparative	E-II
S16	Spracklen et al.	2025	USENIX Security 2025	16 code-generating LLMs	Both	Large-scale	E-II
S17	Fu et al.	2023	arXiv	GitHub Copilot	Western	Empirical	E-II
S18	Gao et al.	2025	arXiv	Multiple	Both	Survey	Secondary

S19	Geruslu et al.	2026	arXiv	Multiple	Both	Synthesis	Secondary
S20	Licorish et al.	2025	arXiv	GPT-4	Western	Comparative	E-II
S21	R. Deng et al.	2024	Computers & Education	ChatGPT	Western	Review + meta-analysis	Secondary
S22	Becker et al.	2023	SIGCSE 2023	ChatGPT, Codex, Copilot	Western	Analysis	Secondary
S23	Smith & Zilles	2024	SIGCSE 2024	GPT-4	Western	Empirical	E-II
S24	Pimenova et al.	2025	arXiv	Multiple (vibe-coding)	General	Qualitative	E-I
S25	Huang et al.	2025	arXiv	AI coding agents	General	Empirical	E-I
S26	Ren et al.	2026	arXiv	Multiple (multilingual)	Both	Comparative	E-II
S27	Raihan et al.	2025	SIGCSE TS 2025	Multiple	Western	Systematic review	Secondary
S28	Umama et al.	2025	IEEE Access	Multiple	Both	Systematic review	Secondary
S29	J. Deng et al.	2026	arXiv	Multiple (vibe-coded apps)	General	Empirical	E-II
S30	Zhao et al.	2025	arXiv	Claude 4 Sonnet (SWE-Agent)	Western	Benchmark	E-II
S31	Churilov	2026	arXiv	Claude Sonnet 4.6, Claude Haiku 4.5, GPT-5.4-mini, Gemini 2.5 Pro, DeepSeek V3.2	Both	Large-scale	E-II
S32	Fawzy et al.	2026	arXiv	Multiple (vibe-coding)	General	Survey	E-I
S33	Uddin	2026	arXiv	Multiple (vibe-coding)	General	Empirical	E-II
S34	Huesca et al.	2026	AI (MDPI)	Generative AI tools	Western	Quasi-experimental	E-I
S35	Guo et al.	2026	Frontiers in Psychology	Generative AI tools	Western	Mixed-methods	E-I
S36	GLM-5 Team	2026	arXiv	GLM-5	Eastern	Technical report	E-III
S37	Cao et al.	2026	arXiv	Qwen3-Coder-Next	Eastern	Technical report	E-III
S38	Kimi Team	2026	arXiv	Kimi K3	Eastern	Technical report	E-III

S39	Rinja et al.	2026	arXiv	Not specified	General	Empirical (classroom)	E-I
S40	Chen et al.	2026	arXiv	Not specified	General	Mixed-methods	E-I
S41	Thorgeirsson et al.	2026	arXiv	Not specified	General	Cross-sectional (preregistered)	E-I
S42	Gama et al.	2025	arXiv (ICSE SEET 2026)	Not specified	General	Empirical (hackathon)	E-I

Note. Independence is assigned from the study design recorded in the Type column, using the tiers defined in Section 3.6. E-I denotes an independent study with human participants; E-II an independent benchmark or audit without human participants; E-III a developer self-report. Secondary denotes a review, survey or synthesis of other studies, admitted for corroboration and context under criterion E4 and never relied on as primary evidence for a dimension outcome. Model version, benchmark and metric are reported in Table 6 and Section 4.4 where the sources state them. Prompt protocol is not tabulated, because the developer technical reports do not disclose it and four of the student-facing studies do not identify their models with sufficient precision to support it.

Publication has accelerated sharply. Four studies appeared in 2023, eleven in 2024 and twelve in 2025. Fifteen have appeared in the first eight months of 2026, so the 2026 total already exceeds any complete preceding year with a third of the year still to run. By model origin, 17 studies evaluated Western models only and seven evaluated Eastern models only. Nine covered both. A further nine adopted a general or model-agnostic approach.

Evidence Independence

The corpus divides unevenly once study design is taken into account. Fourteen studies are independent studies with human participants (E-I), fifteen are independent benchmarks or audits (E-II), seven are developer self-reports (E-III), and six are secondary reviews or surveys admitted for context rather than as primary evidence. That division is not neutral with respect to the comparison this review makes. All seven E-III records evaluate Eastern models, and every Eastern-model record in the corpus is E-III. Not one of the fourteen E-I studies evaluates an Eastern model. The Western and Eastern columns of Table 6 therefore rest on different classes of evidence, and a comparison drawn straight across them compares evaluation regimes as much as it compares models. Comparative statements in Sections 4.4 and 4.5 are accordingly made within a tier wherever the evidence permits. One record alone, Churilov (2026), places models from both blocs under a single independent protocol, which is why it carries a weight in this review out of proportion to its being a single study.

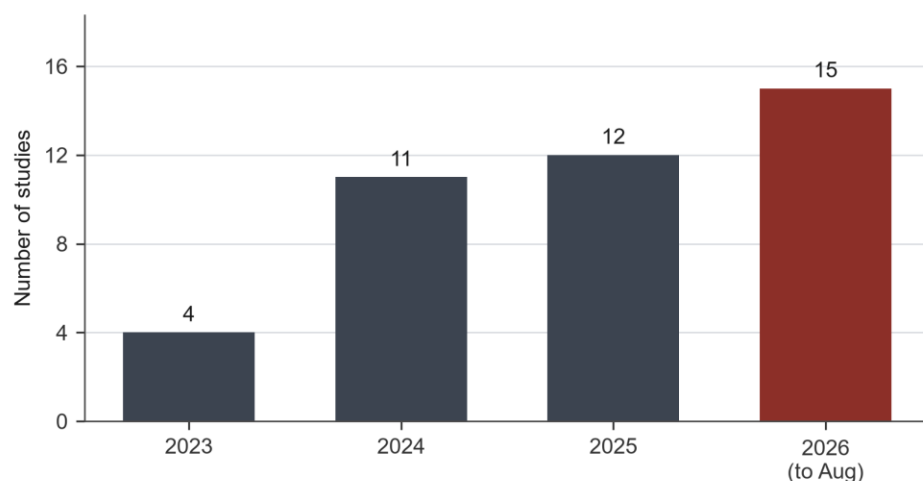


Figure 2 Distribution of included studies by publication year

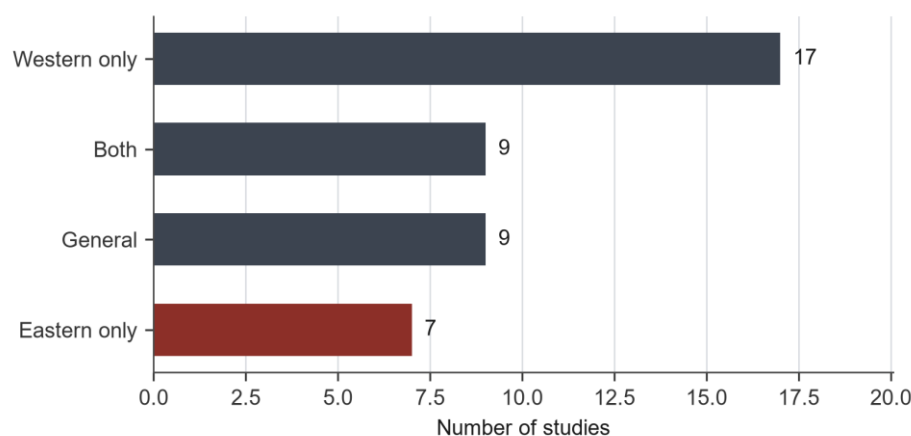


Figure 3 Distribution of included studies by model origin

RQ1: State of Research on Frontier AI Models for Educational Code Generation

Four observations characterise the current state of the field. First, the field is growing rapidly: the 2026 partial year alone contributed fifteen studies, more than any complete preceding year. Second, the research is skewed both geographically and methodologically. Studies of Western models span the full methodological range: empirical comparisons, classroom deployments, quasi-experimental designs and large-scale field trials. Every one of the seven Eastern-model entries, by contrast, is a technical report published by the organisation that built the model (DeepSeek AI, 2024a, 2024b; Hui et al., 2024; Qwen Team, 2025; Cao et al., 2026; GLM-5 Team, 2026; Kimi Team, 2026). The quality appraisal registers this directly: those seven are the only studies in the corpus scoring zero on threats to validity, and all seven sit at the bottom of the quality range. An eighth 2026 release was excluded because its report offers no measurable outcome to assess. Third, education-focused work remains a minority within the broader AI code generation literature. Of the 42 studies reviewed, 16 reported on pedagogical dimensions (D6), and 12 were conducted in educational settings, of which nine were formal courses and three were hackathons or purpose-built student environments. Fourth, publications from 2026 have begun to occupy previously empty ground, particularly in vibe-coding security (J. Deng et al., 2026; Zhao et al., 2025), cross-model hallucination testing (Churilov, 2026) and the study of student vibe-coding practice (Rinja et al., 2026; Thorgeirsson et al., 2026), although the fundamental absence of a cross-bloc classroom study persists.

Author affiliation patterns reinforce this picture. Work on Western models originates predominantly from institutions in the United States, Canada, the United Kingdom and Australia. Technical reports on Eastern models originate from their respective companies in China. Studies that cross the divide, in which researchers from one geography evaluate models from the other, are rare. Umama et al. (2025) observed a comparable pattern across 58 code generation studies, noting that the geography of a research team tends to predict which models it evaluates. In terms of Diffusion of Innovation theory, this reflects the compatibility and observability advantages that Western models hold within English-language research communities.

RQ2: Code Output Comparison Across Dimensions

Table 6 maps the evidence from the included studies against the six comparison dimensions for Western and Eastern models.

Table 6 Code output comparison matrix: Western and Eastern AI models across six dimensions

Dimension	Western Models: Key Evidence	Eastern Models: Key Evidence	Studies
D1: Code correctness	GPT-4: 67.0 HumanEval pass@1 [E-III*]; Claude 3.5 Sonnet: 95% HRI success, Gemini 1.5 Pro 60%,	DeepSeek-R1: accepted solutions in fewer attempts than ChatGPT-o1 [E-II] (Manik, 2025); DeepSeek-V3:	S01-S04, S07, S11-S15, S20,

	ChatGPT 3.5 20% [E-II] (Sobo et al., 2024); GPT-4 comparable to human across 72 SE tasks [E-II] (Licorish et al., 2025); agent-generated code 57% functionally correct [E-II] (Zhao et al., 2025)	65.2 HumanEval pass@1 [E-III] (DeepSeek AI, 2024b); DeepSeek-Coder-V2 reports results superior to GPT-4 Turbo [E-III] (DeepSeek AI, 2024a); Qwen2.5-Coder state of the art on 10+ benchmarks [E-III] (Hui et al., 2024); 2026 releases report state-of-the-art open-model coding results without independent verification [E-III] (GLM-5 Team, 2026; Cao et al., 2026; Kimi Team, 2026)	S23, S30, S33, S36-S38, S40
D2: Code quality	ChatGPT 8.9 min technical debt per function, Copilot 9.1, CodeWhisperer 5.6 with 31.1% correctness [E-II] (Yetiştiren et al., 2023); Copilot-generated code: 29.5% of Python and 24.2% of JavaScript snippets affected across 43 CWE categories [E-II] (Fu et al., 2023); only 11.8% of agent-generated solutions secure [E-II] (Zhao et al., 2025); 180 of 200 audited vibe-coded repositories (90%) contain at least one vulnerability, 76.7% of the 1,471 identified rated critical or high [E-II] (J. Deng et al., 2026)	DeepSeek-R1: stronger maintainability, consistency and structural organisation than ChatGPT-4o across 600 problems [E-II] (Almanasra et al., 2026); Qwen2.5-Coder: cross-file completion [E-III] (Hui et al., 2024)	S01, S02, S12, S15, S17-S20, S29, S30, S32, S40
D3: Hallucination	Commercial models at least 5.2%, open-source up to 21.7% hallucinated packages across 2.23M samples [E-II] (Spracklen et al., 2025); 2026 cohort 4.62% (Claude Haiku 4.5) to 6.10% (GPT-5.4-mini) [E-II] (Churilov, 2026)	DeepSeek V3.2 tested in the 2026 cohort [E-II] and falls within the same range as the four Western models; 127 package names invented identically by all five models, 53 still registrable [E-II] (Churilov, 2026)	S16-S18, S29, S31
D4: Natural language understanding	Limited direct evidence; Western models trained predominantly on English-language data	DeepSeek-Coder-V2: 338 programming languages [E-III] (DeepSeek AI, 2024a); Qwen3: 29 to 119 natural languages and dialects [E-III] (Qwen Team, 2025); Chinese prompts show no efficiency advantage and generally lower success rates than English [E-II] (Ren et al., 2026)	S11, S12, S14, S26
D5: Developer productivity	Copilot: 26.08% more tasks completed than control [E-I] (Ziegler et al., 2024); CodeAid: 79% response accuracy, 86% perceived helpfulness [E-I] (Kazemitabaar et al., 2024); AI-assisted pair programming	DeepSeek-R1: fewer attempts to accepted solution [E-II] (Manik, 2025)	S02, S05, S06, S08, S09, S24, S25, S32, S33

	outperformed individual work [E-I] (Fan et al., 2025)		
D6: Pedagogical appropriateness	CodeAid: 700-student deployment over 12 weeks [E-I] (Kazemitabaar et al., 2024); ChatGPT scaffolding of computational thinking [E-I] (Liao et al., 2024); reduced programming anxiety [E-I] (Fan et al., 2025); 686 students across 53 class groups and 10 campuses [E-I] (Huesca et al., 2026); 83 students, computational thinking [E-I] (Guo et al., 2026); instrumental vs executive help-seeking across 19,418 turns from 110 undergraduates [E-I] (Rinja et al., 2026); writing proficiency and CS achievement predict vibe-coding performance in 100 students [E-I] (Thorgeirsson et al., 2026); code readability assessed on student projects [E-I] (Chen et al., 2026); novice engagement at a university hackathon [E-I] (Gama et al., 2025)	No educational deployment study involving an Eastern model was identified	S05-S08, S10, S21-S23, S27, S34, S35, S39-S42

Note. Bracketed markers record evidence independence, defined in Section 3.6. E-I denotes an independent study with human participants; E-II an independent benchmark or audit without human participants; E-III a developer self-report, in which the organisation that built the model reports its own evaluation of it. E-III* marks a figure for one developer's model reported inside a rival developer's technical report, where the evaluator is independent of the model evaluated but is not disinterested. Comparison between the Western and Eastern columns is sound only within a tier. The D1 row illustrates the asymmetry the review reports: Western correctness evidence is predominantly E-II, Eastern correctness evidence is predominantly E-III.

Code Correctness and Functionality (D1)

Code correctness was the most frequently measured dimension, appearing in 22 of the 42 studies. Four of those 22 are 2026 model technical reports whose correctness claims are self-reported and, for the excluded release, not quantified at all. The evidence indicates marked convergence on standard benchmarks, though not a reversal of position, since DeepSeek-V3 records 65.2 pass@1 on HumanEval against 67.0 for GPT-4 (DeepSeek AI, 2024b), placing the two models within roughly two percentage points of each other. Manik (2025) compared ChatGPT-o1 with DeepSeek-R1 on online judge problems, finding that DeepSeek more frequently achieved an accepted solution on the first attempt while ChatGPT produced fewer code quality issues and shorter solutions.

Correctness on algorithmic benchmarks does not, however, transfer directly to application development. Sobo et al. (2024) evaluated three Western models on code generation for human-robot interaction applications across 20 designed prompts and found pronounced variation: Claude 3.5 Sonnet achieved a 95% success rate, Gemini 1.5 Pro 60% and ChatGPT 3.5 20%. A spread of 75 percentage points was recorded among models from the same geographic bloc. More variance is therefore accounted for by model family and version than by origin. Zhu et al. (2024) reinforced this with DOMAINEVAL, an auto-constructed multi-domain benchmark evaluating 12 representative models across six domains, which found strong performance on computational tasks alongside marked weakness in cryptography and system-level coding.

Licorish et al. (2025) compared GPT-4-generated Python code with human-written code across 72 software engineering tasks and reached a cautious conclusion. The GPT-4 solutions passed more test cases, while human programmers adhered more closely to coding standards and performed better where domain knowledge or unconventional solutions were required. Zhao et al. (2025) introduced a further consideration, benchmarking agent-generated code on 186 feature-request tasks drawn from real-world open-source projects. Of the solutions produced by SWE-Agent with Claude 4 Sonnet, 57% were functionally correct and only 11.8% were secure. D1 and D2 are separated by that gap far more sharply than benchmark scores alone suggest.

Code Quality and Maintainability (D2)

Ten studies addressed code quality. Yetiştiren et al. (2023) measured technical debt across three Western tools, reporting average values of 8.9 minutes per function for ChatGPT, 9.1 minutes for GitHub Copilot and 5.6 minutes for Amazon CodeWhisperer. The comparatively low figure for CodeWhisperer accompanies a lower correctness rate of 31.1%, which suggests a trade-off between correctness and maintainability rather than a uniform quality ranking.

Among Eastern models, Almanasra et al. (2026) evaluated 600 Java and Python problems and found that DeepSeek-R1 produced code with stronger maintainability, greater consistency and better structural organisation than ChatGPT-4o, assessed against code structure, method and class design, naming conventions, error handling and documentation. Qwen2.5-Coder was reported by Hui et al. (2024) to perform well on cross-file code completion. That task requires contextual awareness across an entire project.

The vibe-coding security literature published in 2026 extended this dimension considerably. J. Deng et al. (2026) audited 200 deployed vibe-coded applications, sampled from 1,170 deployed web applications within a corpus of 10,517, combining agent-assisted code auditing with human validation. They identified 1,471 vulnerabilities, of which 20% were rated critical and 56.7% high, against 2.2% rated low. Of the 200 repositories audited, 180 (90%) contained at least one vulnerability. The vulnerabilities followed recurring patterns that differ from those of conventional development workflows, including placeholder logic, unfiltered input and secret exposure, concentrated in backend code, access control and cryptographic handling. The authors attribute these to systematic limitations of AI agents across the vibe-coding lifecycle, among them memory loss, locally optimised objectives and insufficient security knowledge, and conclude that improvements in model capability and prompting strategy reduce the underlying risk without eliminating it. Fu et al. (2023) had earlier identified security weaknesses in Copilot-generated code within GitHub projects, affecting 29.5% of Python and 24.2% of JavaScript snippets across 43 weakness categories, eight of which appear in the 2023 CWE Top 25. Code quality has begun to be assessed in educational settings as well. Chen et al. (2026) scored student hackathon projects on code readability alongside functionality, under a rule requiring participants to use generated code without manual editing. It is the only study in the corpus where a quality criterion is applied to student-produced vibe-coded work. Read through the Technological Pedagogical Content Knowledge framework, these findings indicate that what a model can produce is insufficient without the pedagogical knowledge required to teach students to scrutinise and verify what it generates.

Hallucination and Fabrication (D3)

Five studies addressed hallucination, with the 2026 literature contributing a further advance. Spracklen et al. (2025) generated 2.23 million code samples across 16 models and reported average hallucinated package rates of at least 5.2% for commercial models and 21.7% for open-source models.

Churilov (2026) tested five frontier models released between October 2025 and March 2026, including an Eastern model for the first time. Across 199,845 paired Python and JavaScript prompts validated against the PyPI and npm master lists, hallucination rates ranged from 4.62% for Claude Haiku 4.5 to 6.10% for GPT-5.4-mini. DeepSeek V3.2 fell within that band. It was placed neither clearly better nor worse than its Western counterparts. The study also identified 127 package names invented identically by all five models, of which 53 remained

registrable by a third party, indicating that hallucination patterns may derive from shared training data artefacts rather than from anything specific to a model family. The finding is notable here because it constitutes independent third-party evidence placing an Eastern model directly alongside Western models under identical conditions.

For educational settings, the implications are serious, because students learning to program frequently lack the experience needed to distinguish a hallucinated import from a legitimate one. The associated costs fall disproportionately on beginners, and they include wasted debugging time, mislearned patterns and potential security exposure.

Natural Language Understanding (D4)

Five studies addressed the multilingual dimension. DeepSeek-Coder-V2 supports 338 programming languages (DeepSeek AI, 2024a), and Qwen3 expanded natural language support from 29 to 119 languages and dialects (Qwen Team, 2025). These are engineering achievements. Breadth of language coverage is addressed, rather than how well a natural language prompt is comprehended.

Ren et al. (2026) tested whether Chinese-language prompts produce better results than English prompts for vibe-coding and found no efficiency advantage for Chinese. Token consumption varied unpredictably by model, and the success rate when prompting in Chinese was generally lower than in English across every model tested. The finding challenges a widely held assumption and carries direct weight for educators choosing between Chinese-language and English-language AI tools.

RQ3: Productivity and Pedagogical Impact

Developer Productivity (D5)

Ten studies addressed productivity, and the strongest comes from Ziegler et al. (2024), who measured the impact of GitHub Copilot in workplace environments across three companies, including Microsoft and Accenture, and reported that the Copilot group completed 26.08% more tasks than the control group.

In educational settings, Kazemitabaar et al. (2024) reported 86% perceived helpfulness and 79% response accuracy for CodeAid over a twelve-week deployment. Fan et al. (2025) found that AI-assisted pair programming outperformed individual work on task completion in a two-year study with 234 undergraduates, with results comparable to human-human pairing.

Productivity data for Eastern models remain limited. Manik (2025) found DeepSeek producing accepted solutions in fewer attempts than ChatGPT-o1, but this measures code-level efficiency rather than workflow productivity. Fawzy et al. (2026) surveyed 162 vibe coders across three experience groups comprising non-coders, novices and professional developers, and found that motivations, interaction styles and quality assurance practices diverge with experience while perceptions of code quality remain broadly similar. They characterise this as a perception-action gap, in which awareness of the risks in AI-generated code is broadly distributed but the capacity to evaluate, debug and verify remains experience-dependent. Neither this study nor Uddin (2026) compared specific Western and Eastern models against one another.

Pedagogical Appropriateness (D6)

Sixteen studies addressed pedagogical dimensions, making D6 the second most frequently covered dimension after code correctness. The evidence, however, draws exclusively on Western or unidentified models. R. Deng et al. (2024) conducted a systematic review and meta-analysis of 69 experimental studies of the effect of ChatGPT on student learning, reporting improvements in academic performance, affective-motivational states and higher-order thinking, a reduction in mental effort, and no significant effect on self-efficacy. Becker et al. (2023) argued

that AI code generation alters the nature of programming assignments and that educators must redesign how student work is assessed. Huesca et al. (2026) provided the largest-sample evidence available, testing four generative AI-supported instructional strategies across 686 students in a pre-test and post-test design. Guo et al. (2026) contributed mixed-methods evidence from 83 tenth-grade students on computational thinking.

The 2026 literature has begun to examine vibe-coding practice among students directly. Rinja et al. (2026) analysed 19,418 interaction turns from 110 undergraduates and distinguished instrumental help-seeking, characterised by inquiry and exploration and associated with higher performance, from executive help-seeking, characterised by task delegation and associated with lower performance. Thorgeirsson et al. (2026) ran a preregistered cross-sectional study of 100 tertiary students and found that written-communication proficiency and computer science achievement both predicted vibe-coding performance, with the latter remaining significant after controlling for domain-general cognitive ability. Chen et al. (2026) evaluated a month-long multi-track hackathon in which participants were required to use generated code without manual editing, and Gama et al. (2025) observed 31 undergraduates across nine teams in a one-day hackathon at a Brazilian public university. None of these four studies identifies the models used with enough precision to attribute outcomes to a specific system, which is why each scores zero or 0.5 on QA2.

The absence of any educational deployment study involving an Eastern model is the most conspicuous gap identified by this review, and the update search did not close it. The four student-facing studies added in 2026 either use Western models or do not identify the model at all. In the terms of the Technological Pedagogical Content Knowledge framework, the technological knowledge of Eastern models is partially characterised through D1, D2 and D4, whereas their performance at the intersection where technology, pedagogy and content knowledge converge is entirely undocumented.

RQ4: Gaps and Future Directions

Four structural gaps emerge from the synthesis. First, no cross-bloc classroom study exists. Among the 42 studies reviewed, only four compared Western and Eastern models directly, and none did so in a formal educational setting. Second, self-evaluation bias is pervasive and has not diminished with scale. All seven Eastern model technical reports in the corpus were published by the organisations that built the models, including the 2026 frontier releases, and independent replication by third-party researchers is required. Third, the hallucination evidence base is narrowing but remains thin. Churilov (2026) represents a genuine advance in placing DeepSeek V3.2 alongside four Western models, but it remains a single study, and other Eastern models including Qwen, GLM and Kimi have not been tested for hallucination rates. Fourth, capability reporting at the frontier is deteriorating rather than improving. One flagship 2026 release could not be included at all because its technical report states no measurable capability outcome, while commercial aggregators circulate benchmark scores for that same model differing by more than ten percentage points. A review that depends on reported evidence cannot assess a model documented this way.

Fifth, vibe-coding security is an area of growing concern. J. Deng et al. (2026) found that 90% of 200 audited vibe-coded repositories contained at least one vulnerability, with more than three-quarters of the 1,471 vulnerabilities identified rated critical or high severity, and Zhao et al. (2025) found only 11.8% of functionally correct agent-generated solutions to be secure. This carries direct implications for educational settings in which students may deploy vibe-coded applications without security review.

Thematic Analysis

Six themes were identified through the reflexive thematic analysis. Table 7 summarises each theme, the contributing studies and the key evidence.

Table 7 Thematic analysis summary

Theme	Description	Studies	Key Evidence
T1: Performance convergence	Eastern models perform within the range established by Western models on code correctness	S01, S02, S04, S11, S13, S31, S36-S38	DeepSeek-V3 at 65.2 HumanEval against GPT-4 at 67.0; DeepSeek-Coder-V2 reports results superior to GPT-4 Turbo; DeepSeek V3.2 hallucination rate within the Western band; 2026 releases claim frontier parity on vendor-reported figures
T2: Language and localisation barriers	Prompt language affects output; Chinese is not inherently more efficient	S11, S12, S14, S26	Chinese prompts show no cost advantage and generally lower success rates (Ren et al., 2026); Qwen3 supports 119 languages
T3: Benchmark validity concerns	Standard benchmarks are saturating; real-world and security benchmarks are needed	S04, S15, S20, S28, S29, S30	HumanEval approaching ceiling effects; 57% of agent-generated solutions correct but only 11.8% secure (Zhao et al., 2025)
T4: Training data opacity	Training data are rarely disclosed; hallucination patterns may reflect shared data	S11, S13, S14, S16, S31, S36-S38	127 package names invented identically across five models drawn from both blocs (Churilov, 2026); training corpora undisclosed across all seven Eastern technical reports in the corpus
T5: Access, cost and regulatory asymmetry	Eastern models are frequently cheaper or open-weight but face export controls	S11, S13, S14, S24, S29, S36-S38	DeepSeek models released under permissive licences at lower cost; open-weight releases continue through the 2026 cohort; Chinese AI regulation differs from United States frameworks
T6: Pedagogical model lock-in	Educational research is built exclusively on Western models, creating path dependency	S05-S08, S10, S21, S27, S34, S35, S39-S42	Every classroom study in the corpus uses a Western model or does not identify the model; consistent with Diffusion of Innovation theory

Theme T6 warrants closer examination. Viewed through Diffusion of Innovation theory (Rogers, 2003), the pattern is predictable, since Western models entered the market earlier, which conferred relative advantage. They aligned more readily with English-language research conventions, conferring compatibility, and achieved greater visibility in academic venues, conferring observability. The result is a self-reinforcing loop. Educators adopt tools that have been tested, researchers test the tools educators have adopted, and alternatives are left unexamined. For institutions in regions where Eastern models may be more accessible, more affordable or more linguistically appropriate, that loop narrows the range of defensible options.

DISCUSSION

Summary of Key Findings

The synthesis of 42 studies across six dimensions and two theoretical lenses supports five propositions, none of them straightforward. The first concerns convergence. On code correctness benchmarks, the distance between the

two groups has narrowed to the point of overlapping performance ranges (Theme T1). DeepSeek-V3 records 65.2 pass@1 on HumanEval against 67.0 for GPT-4, a difference of under two percentage points, and DeepSeek-R1 achieves accepted solutions in fewer attempts than ChatGPT-o1 on algorithmic problems (Manik, 2025). Neither Eastern nor Western superiority is established by these results. Sobo et al. (2024) recorded a spread of 75 percentage points among three Western models on a single human-robot interaction task, with Claude 3.5 Sonnet at 95% and ChatGPT 3.5 at 20%, so model family and version account for more of the observed variance than geographic origin does.

The second concerns asymmetry. Western models have been subjected to the full range of evaluation methods, encompassing benchmarks, classroom deployments, quasi-experimental designs, large-scale field trials and independent security audits. Eastern models have been assessed principally on D1, occasionally on D4, and exclusively, across all seven entries, in reports authored by the organisations that developed them (Theme T6). This did not change as the models advanced. Releases from 2026 are documented in precisely the same way as their 2024 predecessors, and one could not be assessed at all. The two groups are comparably matched on benchmarks. They are not comparably matched on evidence.

The third concerns hallucination. Churilov (2026) measured rates between 4.62% and 6.10% across five frontier models, including DeepSeek V3.2, and found that all five invented the same 127 package names, 53 of which remained registrable by a third party. The convergence of error patterns across models from both blocs suggests a common origin in shared training data rather than in the design of any particular model family. This result carries particular evidential weight because it is independent third-party testing under identical conditions, which is precisely what the Eastern model literature otherwise lacks. It remains, however, a single study.

The fourth concerns security, and Zhao et al. (2025) found that 57% of agent-generated solutions to 186 real-world feature-request tasks were functionally correct while only 11.8% of them were secure. J. Deng et al. (2026) audited 200 deployed vibe-coded applications and found that 180 of them, or 90%, contained at least one vulnerability; of the 1,471 vulnerabilities identified, 20% were critical and 56.7% high severity. The patterns were specific to vibe-coded applications, including placeholder logic, unfiltered input and secret exposure, and traced to systematic limitations of AI agents across the development lifecycle. For professional teams operating a security review pipeline, this constitutes a workflow problem. In an educational setting, it becomes a pedagogical hazard, because functional correctness is the property a novice can readily observe and security is the property a novice cannot.

The fifth concerns the maturation of vibe-coding research itself. Twelve studies in the corpus address the practice directly, four of them predating 2026, and the emphasis has shifted from perception surveys towards direct measurement. Students are examined specifically in five of the twelve. That category had no representation before 2025.

Implications for Educators and Developers

For a programming lecturer in Southeast Asia, the principal implication concerns the basis on which a tool is selected rather than which tool is selected. No Eastern model has been evaluated in a classroom setting. That does not indicate such models are unsuitable for classroom use; it indicates that the evidence required to make the judgement does not exist. Lecturers defaulting to ChatGPT or GitHub Copilot are selecting on the basis of available research rather than demonstrated comparative superiority, and the distinction matters where institutional procurement decisions are justified by reference to an evidence base. A structured means of approaching the decision is offered by the Technological Pedagogical Content Knowledge framework, under which the question is not only whether the model generates correct code (D1 to D4) but whether it aligns with instructional goals and classroom workflow (D5 and D6). A model that performs well on every benchmark while producing code it cannot explain has limited pedagogical value.

On productivity, the evidence is more concrete. Ziegler et al. (2024) measured a 26.08% increase in completed tasks among Copilot users across workplaces including Microsoft and Accenture. Fan et al. (2025) found that AI-assisted pair programming reduced programming anxiety and improved intrinsic motivation relative to individual work across 234 undergraduates over two years, with outcomes comparable to human-human pairing. These findings support a place for such tools in programming pedagogy, and direct attention to the question of implementation. CodeAid, as described by Kazemitabaar et al. (2024), offers one indication: the system deliberately withholds direct code solutions in favour of pseudocode and conceptual explanation. That design decision, which embeds pedagogical restraint within the tool itself, may account for much of its effectiveness, and it indicates that AI assistance is not uniform in its educational value.

Developers building educational applications face a distinct problem. Only 11.8% of functionally correct agent-generated solutions were assessed as secure in a benchmark of real-world tasks (Zhao et al., 2025), and 90% of deployed vite-coded repositories audited by J. Deng et al. (2026) carried at least one vulnerability, more than three-quarters of them critical or high severity. These arise from limitations that improved prompting reduces without eliminating. A professional team with a security review pipeline can absorb this. A student deploying a class project generally cannot and is unlikely to recognise the exposure.

The East-West Divide: Convergence or Divergence?

The evidence supports both readings simultaneously, and the apparent contradiction is the substantive finding of this review. On capability, the pattern is convergence. DeepSeek-V3 performs within two percentage points of GPT-4 on HumanEval, DeepSeek-Coder-V2 reports results superior to GPT-4 Turbo on coding benchmarks, and the Qwen2.5-Coder series reports state-of-the-art performance across more than ten coding evaluations. Where the question concerns which model writes better code, the answer depends on the task rather than on the origin of the model. On the evidence, the pattern is divergence, and the gap is wide. Western models benefit from a research ecosystem that evaluates them in classrooms, submits them to independent security audits and publishes the results in peer-reviewed venues. Eastern models benefit from a market ecosystem that prioritises rapid deployment and open-weight release. Neither ecosystem is deficient in itself; they optimise for different objectives. The consequence, however, is that a lecturer choosing between GitHub Copilot and CodeGeeX has a large body of published research supporting the first option and none supporting the second. This is a visibility gap rather than a quality gap, and Diffusion of Innovation theory anticipates it.

Two developments would alter this position. The first is independent evaluation of Eastern models, conducted by researchers unconnected to their development, using the instruments already applied to Western models. Churilov (2026) demonstrates that such work is feasible and informative. The second is classroom experimentation placing a Western model and an Eastern model within the same course, with the same students and the same assessment rubric. The Technological Pedagogical Content Knowledge framework provides the design basis for such studies, and Diffusion of Innovation theory explains why they have not yet been undertaken.

The Moving Frontier

A structural difficulty attends any systematic review in this field. Model releases outpace the evidence about them, and the gap is not closing. The update search brought part of the 2026 frontier inside the corpus, since GLM-5, Qwen3-Coder-Next and Kimi K3 published technical reports within the review window. It could not bring in any independent assessment of them, and in one case it could not bring in the model at all. The DeepSeek-V4 preview report states no benchmark, no dataset and no metric, only architecture and inference efficiency, so a review working from reported evidence has nothing to assess. The model is widely held to be among the strongest open-weight coding systems available; that belief rests on figures no peer-reviewed or preprint source states. Every claim about these models originates either with the organisation that sells the model or with a commercial aggregator that does not disclose its methodology. Those aggregators disagree with one another. The SWE-bench

Verified score circulating for DeepSeek V4 varies by more than ten percentage points depending on which is consulted.

Aggregator figures fall within exclusion criteria E1 and E5, and no claim in this review rests upon them. The vendor technical reports that do meet the criteria are included, but the appraisal instrument records what that inclusion costs: all seven score zero on threats to validity, and all seven sit at the bottom of the quality range. The most capable models available at the time of writing are documented in the corpus without being independently evaluated in it. This is the evidence asymmetry of Theme T4 observed at the frontier rather than in retrospect, and it applies to Western and Eastern models alike. The problem identified by this review is not a transient artefact of publication lag. It is a structural feature of how frontier model capability is currently reported, and the 2026 cohort reproduces it exactly.

LIMITATIONS

Several limitations qualify the findings reported here. First, the search was conducted through web-based database queries rather than direct institutional database access, which may have excluded studies published in Chinese-language venues without English abstracts. Second, the thin-literature strategy broadened inclusion to general software development, so not every study in the corpus is education-focused. Third, preprints and developer technical reports are heavily represented in the corpus, particularly for Eastern models and for the 2026 vibecoding literature. This follows from the eligibility rule set out in Section 3.3 rather than from an oversight, and many of the preprints have since been accepted at established venues. The consequence nonetheless stands: a substantial part of the evidence base has not been externally refereed, and the appraisal instrument constrains that evidence without supplying the scrutiny peer review would have applied. Fourth, the pace of model development means that benchmarks published in 2024 may have limited bearing on the capabilities of models released in 2026, as Section 5.4 sets out. Fifth, a formal inter-rater agreement coefficient was not computed for the screening process. Sixth, the initial search under-retrieved: an update search conducted on 1 September 2026 identified nine in-window records it had missed, including a model technical report whose title contains the review's own primary search term. The update is documented in Section 3.2 and its yield is included in the corpus, but the omission indicates that the database queries alone were not sufficient and that citation chasing carried more weight than anticipated. Seventh, no claim to completeness is made for a field that continues to expand rapidly.

A further qualification applies to the comparison itself. Model origin is not an experimental variable, and the studies grouped under each heading differ in model generation and version, task, benchmark, dataset, prompting strategy, sample size and evaluation protocol. Any of these may account for a difference attributed here to geography. This review therefore treats origin as a proxy for membership of an evidence ecosystem rather than as a cause of output quality, and the data support that reading. Sobo et al. (2024) recorded a spread of 75 percentage points among three models drawn from the same bloc, which is wider than any difference this review observes between blocs. Comparative statements are accordingly made within study design and independence tiers rather than across them, and the convergence reported as Theme T1 should be read as the absence of a demonstrated origin effect rather than as a demonstration of equivalence.

CONCLUSION

This review examined 42 studies published between January 2022 and August 2026. It addressed a question the literature had not previously posed directly: how the code outputs of Western and Eastern AI models compare when used for educational application development. The Technological Pedagogical Content Knowledge framework and Diffusion of Innovation theory provided the theoretical lenses, PRISMA 2020 provided the methodology, and each study was assessed against a six-dimension framework developed for this purpose.

The findings are twofold. On capability, the two groups have converged, since Eastern models perform within the range established by their Western counterparts on standard coding benchmarks, and the hallucination evidence

reported by Churilov (2026) indicates that even the error patterns are shared. On the evidence, no convergence has occurred at all. Western models have been tested in classrooms, examined by independent researchers and assessed for security; Eastern models have not, and the 2026 frontier releases repeat the pattern rather than breaking it. The field has documented what GPT-based tools do to student programming performance across a semester. It has documented nothing equivalent for DeepSeek.

This work makes four contributions. The first is a systematic synthesis, believed to be the first of its kind, framing AI code generation research around the Western and Eastern distinction and mapping both the capability convergence and the evidence divergence. The second is a theoretical grounding. What should be measured is specified by the Technological Pedagogical Content Knowledge framework, and why it has not been is explained by Diffusion of Innovation theory. The third is a six-dimension framework covering correctness, quality, hallucination, natural language understanding, productivity and pedagogical appropriateness, designed for reuse in comparative studies beyond this review. The fourth is the most consequential: a research gap that shows no sign of closing, since no study has evaluated Western and Eastern models side by side in a formal educational setting.

The practical recommendation follows directly. For educators in Malaysia, Indonesia, Thailand and other regions served by both ecosystems, benchmark results are an insufficient basis for determining which model belongs in a classroom. Both should be piloted, both should be measured, and the results should be published, since the field requires such data more than it requires additional benchmark leaderboards. For researchers seeking the single most valuable next study, this review nominates a controlled classroom experiment comparing a Western model with an Eastern model within the same course, with the same students and the same assessment rubric, scored across the six dimensions proposed here.

The use of these tools by students is no longer in question. What remains in question is whether the research community, the teaching profession and those responsible for policy will generate the evidence required for that use to be informed.

ACKNOWLEDGEMENTS

The authors thank the research collaborators and co-authors for the assistance and expertise shared throughout the review process and for serving as a second screener during the study selection and analysis phases.

DECLARATIONS

The authors declare no conflicts of interest. This research received no specific grant from any funding agency in the public, commercial or not-for-profit sectors. The data extracted from the included studies, comprising the study characteristics table, the six-dimension comparison matrix and the thematic summary, are presented in full in Tables 5 to 7 and in Appendix Table A1. Quality appraisal used the instrument set out in Table 3; summary scores by study type are reported in Section 4.1. No additional datasets were generated.

REFERENCES

1. Almanasra, S., Suwais, K., & AlRamouni, S. (2026). A comparative evaluation of ChatGPT-4o and DeepSeek-R1 for AI-generated code quality using software engineering metrics. *Array*, 30, Article 100979. <https://doi.org/10.1016/j.array.2026.100979>
2. Becker, B. A., Denny, P., Finnie-Ansley, J., Luxton-Reilly, A., Prather, J., & Santos, E. A. (2023). Programming is hard, or at least it used to be: Educational opportunities and challenges of AI code generation. In *Proceedings of the 54th ACM Technical Symposium on Computer Science Education V. 1* (pp. 500–506). Association for Computing Machinery. <https://doi.org/10.1145/3545945.3569759>

3. Braun, V., & Clarke, V. (2006). Using thematic analysis in psychology. *Qualitative Research in Psychology*, 3(2), 77–101. <https://doi.org/10.1191/1478088706qp063oa>
4. Campbell, M., McKenzie, J. E., Sowden, A., Katikireddi, S. V., Brennan, S. E., Ellis, S., Hartmann-Boyce, J., Ryan, R., Shepperd, S., Thomas, J., Welch, V., & Thomson, H. (2020). Synthesis without meta-analysis (SWiM) in systematic reviews: Reporting guideline. *BMJ*, 368, l6890. <https://doi.org/10.1136/bmj.l6890>
5. Cao, R., Chen, M., Chen, J., Cui, Z., Feng, Y., Hui, B., Jing, Y., Li, K., Li, M., Lin, J., Ma, Z., Shum, K., Wang, X., Wei, J., Yang, J., Zhang, J., Zhang, L., Zhang, Z., Zhao, W., & Zhou, F. (2026). Qwen3-Coder-Next technical report [Preprint]. arXiv. <https://doi.org/10.48550/arXiv.2603.00729>
6. Chen, A. J., Cao, Y., Shao, M., Karri, R., & Shafique, M. (2026). Code for all: Educational applications of the “vibe coding” hackathon in programming education across all skill levels [Preprint]. arXiv. <https://doi.org/10.48550/arXiv.2604.22747>
7. Churilov, A. (2026). The range shrinks, the threat remains: Re-evaluating LLM package hallucinations on the 2026 frontier-model cohort [Preprint]. arXiv. <https://doi.org/10.48550/arXiv.2605.17062>
8. DeepSeek AI. (2024a). DeepSeek-Coder-V2: Breaking the barrier of closed-source models in code intelligence [Preprint]. arXiv. <https://doi.org/10.48550/arXiv.2406.11931>
9. DeepSeek AI. (2024b). DeepSeek-V3 technical report [Preprint]. arXiv. <https://doi.org/10.48550/arXiv.2412.19437>
10. Deng, J., Fan, Z., & Meng, R. (2026). Understanding the (in)security of vibe-coded applications [Preprint]. arXiv. <https://doi.org/10.48550/arXiv.2606.23130>
11. Deng, R., Jiang, M., Yu, X., Lu, Y., & Liu, S. (2024). Does ChatGPT enhance student learning? A systematic review and meta-analysis of experimental studies. *Computers & Education*, 227, 105224. <https://doi.org/10.1016/j.compedu.2024.105224>
12. Fan, G., Liu, D., Zhang, R., & Pan, L. (2025). The impact of AI-assisted pair programming on student motivation, programming anxiety, collaborative learning, and programming performance: A comparative study with traditional pair programming and individual approaches. *International Journal of STEM Education*, 12(1), Article 16. <https://doi.org/10.1186/s40594-025-00537-3>
13. Fawzy, A., Tahir, A., & Blincoe, K. (2026). From prompting to verification: How experience shapes vibe coding practices [Preprint]. arXiv. <https://doi.org/10.48550/arXiv.2605.24521>
14. Fu, Y., Liang, P., Tahir, A., Li, Z., Shahin, M., Yu, J., & Chen, J. (2023). Security weaknesses of Copilot-generated code in GitHub projects: An empirical study [Preprint]. arXiv. <https://doi.org/10.48550/arXiv.2310.02059>
15. Gama, K., Calegario, F., Jackson, V., Nolte, A., Morais, L. A., & Garcia, V. (2025). “Can you feel the vibes?”: An exploration of novice programmer engagement with vibe coding [Preprint]. arXiv. <https://doi.org/10.48550/arXiv.2512.02750>
16. Gao, R., Tahir, A., Liang, P., Susnjak, T., & Khomh, F. (2025). A survey of bugs in AI-generated code [Preprint]. arXiv. <https://doi.org/10.48550/arXiv.2512.05239>
17. Geruslu, V., Aliyeva, Z., & Tüzün, E. (2026). Factors influencing the quality of AI-generated code: A synthesis of empirical evidence [Preprint]. arXiv. <https://doi.org/10.48550/arXiv.2603.25146>
18. GLM-5 Team. (2026). GLM-5: From vibe coding to agentic engineering [Preprint]. arXiv. <https://doi.org/10.48550/arXiv.2602.15763>
19. Guo, R., Li, G., Miao, H., Pi, Z., & Xie, L. (2026). Impact of generative AI-assisted programming on the computational thinking of high school students. *Frontiers in Psychology*, 17, Article 1703177. <https://doi.org/10.3389/fpsyg.2026.1703177>
20. Huang, R., Reyna, A., Lerner, S., Xia, H., & Hempel, B. (2025). Professional software developers don't vibe, they control: AI agent use for coding in 2025 [Preprint]. arXiv. <https://doi.org/10.48550/arXiv.2512.14012>
21. Huesca, G., Martinez-Trevino, Y., Jiménez González, C. G., Cantú Delgado, D. A., Navarrete, C., Cedillo-Hernandez, A., & Quintero Meza, R. R. (2026). Teaching programming in the age of generative artificial intelligence: Learning gains and pedagogical integration in a higher education context. *AI*, 7(7), Article 248. <https://doi.org/10.3390/ai7070248>

22. Hui, B., Yang, J., Cui, Z., Yang, J., Liu, D., Zhang, L., Liu, T., Zhang, J., Yu, B., Lu, K., Dang, K., Fan, Y., Zhang, Y., Yang, A., Men, R., Huang, F., Zheng, B., Miao, Y., Quan, S., ... Lin, J. (2024). Qwen2.5-Coder technical report [Preprint]. arXiv. <https://doi.org/10.48550/arXiv.2409.12186>
23. Karpathy, A. [@karpathy]. (2025, February 2). There's a new kind of coding I call 'vibe coding', where you fully give in to the vibes, embrace exponentials, and forget that the code even exists [Post]. X. <https://x.com/karpathy/status/1886192184808149383>
24. Kazemitabaar, M., Ye, R., Wang, X., Henley, A. Z., Denny, P., Craig, M., & Grossman, T. (2024). CodeAid: Evaluating a classroom deployment of an LLM-based programming assistant that balances student and educator needs. In Proceedings of the 2024 CHI Conference on Human Factors in Computing Systems. Association for Computing Machinery. <https://doi.org/10.1145/3613904.3642773>
25. Kimi Team. (2026). Kimi K3: Open frontier intelligence [Preprint]. arXiv. <https://doi.org/10.48550/arXiv.2607.24653>
26. Liao, J., Zhong, L., Zhe, L., Xu, H., Liu, M., & Xie, T. (2024). Scaffolding computational thinking with ChatGPT. IEEE Transactions on Learning Technologies, 17, 1628–1642. <https://doi.org/10.1109/TLT.2024.3392896>
27. Licorish, S. A., Bajpai, A., Arora, C., Wang, F., & Tantithamthavorn, K. (2025). Comparing human and LLM generated code: The jury is still out! [Preprint]. arXiv. <https://doi.org/10.48550/arXiv.2501.16857>
28. Manik, M. M. H. (2025). ChatGPT vs. DeepSeek: A comparative study on AI-based code generation [Preprint]. arXiv. <https://doi.org/10.48550/arXiv.2502.18467>
29. Mishra, P., & Koehler, M. J. (2006). Technological pedagogical content knowledge: A framework for teacher knowledge. Teachers College Record, 108(6), 1017–1054. <https://doi.org/10.1111/j.1467-9620.2006.00684.x>
30. Page, M. J., McKenzie, J. E., Bossuyt, P. M., Boutron, I., Hoffmann, T. C., Mulrow, C. D., Shamseer, L., Tetzlaff, J. M., Akl, E. A., Brennan, S. E., Chou, R., Glanville, J., Grimshaw, J. M., Hróbjartsson, A., Lalu, M. M., Li, T., Loder, E. W., Mayo-Wilson, E., McDonald, S., ... Moher, D. (2021). The PRISMA 2020 statement: An updated guideline for reporting systematic reviews. BMJ, 372, Article n71. <https://doi.org/10.1136/bmj.n71>
31. Pimenova, V., Fakhoury, S., Bird, C., Storey, M.-A., & Endres, M. (2025). Good vibrations? A qualitative study of co-creation, communication, flow, and trust in vibe coding [Preprint]. arXiv. <https://doi.org/10.48550/arXiv.2509.12491>
32. Qwen Team. (2025). Qwen3 technical report [Preprint]. arXiv. <https://doi.org/10.48550/arXiv.2505.09388>
33. Raihan, N., Siddiq, M. L., Santos, J. C. S., & Zampieri, M. (2025). Large language models in computer science education: A systematic literature review. In Proceedings of the 56th ACM Technical Symposium on Computer Science Education V. 1. Association for Computing Machinery. <https://doi.org/10.1145/3641554.3701863>
34. Ren, S., Shen, X., Zhou, Y., Ng, D., & Raj, A. (2026). Chinese language is not more efficient than English in vibe coding: A preliminary study on token cost and problem-solving rate [Preprint]. arXiv. <https://doi.org/10.48550/arXiv.2604.14210>
35. Rinja, D., Oliveira, E. A., López-Pernas, S., Saqr, M., Specht, M., & Misiejuk, K. (2026). Unpacking vibe coding: Help-seeking processes in student-AI interactions while programming [Preprint]. arXiv. <https://doi.org/10.48550/arXiv.2604.27134>
36. Rogers, E. M. (2003). Diffusion of innovations (5th ed.). Free Press.
37. Smith, D. H., IV, & Zilles, C. (2024). Evaluating large language model code generation as an autograding mechanism for “Explain in Plain English” questions. In Proceedings of the 55th ACM Technical Symposium on Computer Science Education V. 2. Association for Computing Machinery. <https://doi.org/10.1145/3626253.3635542>
38. Sobo, A., Mubarak, A., Baimagambetov, A., & Polatidis, N. (2024). Evaluating LLMs for code generation in HRI: A comparative study of ChatGPT, Gemini, and Claude. Applied Artificial Intelligence, 39(1), Article 2439610. <https://doi.org/10.1080/08839514.2024.2439610>
39. Spracklen, J., Wijewickrama, R., Sakib, A. H. M. N., Maiti, A., Viswanath, B., & Jadliwala, M. (2025). We have a package for you! A comprehensive analysis of package hallucinations by code generating

- LLMs. In Proceedings of the 34th USENIX Security Symposium. USENIX Association. <https://www.usenix.org/conference/usenixsecurity25/presentation/spracklen>
40. Sun, D., Boudouaia, A., Zhu, C., & Li, Y. (2024). Would ChatGPT-facilitated programming mode impact college students' programming behaviors, performances, and perceptions? An empirical study. *International Journal of Educational Technology in Higher Education*, 21, Article 14. <https://doi.org/10.1186/s41239-024-00446-5>
 41. Thorgeirsson, S., Weidmann, T. B., & Su, Z. (2026). Computer science achievement and writing skills predict vibe coding proficiency [Preprint]. arXiv. <https://doi.org/10.48550/arXiv.2603.14133>
 42. Uddin, S. M. J. (2026). Is vibe coding the future? An empirical assessment of LLM generated codes for construction safety [Preprint]. arXiv. <https://doi.org/10.48550/arXiv.2604.12311>
 43. Umama, Danyaro, K. U., Nasser, M., Zakari, A., Abdullahi, S., Khanzada, A., Yakubu, M. M., & Shoaib, S. (2025). LLM-based code generation: A systematic literature review with technical and demographic insights. *IEEE Access*, 13, 194915–194939. <https://doi.org/10.1109/ACCESS.2025.3631952>
 44. Wermelinger, M. (2023). Using GitHub Copilot to solve simple programming problems. In Proceedings of the 54th ACM Technical Symposium on Computer Science Education V. 1 (pp. 172–178). Association for Computing Machinery. <https://doi.org/10.1145/3545945.3569830>
 45. Yetiştiren, B., Özsoy, I., Ayerdem, M., & Tüzün, E. (2023). Evaluating the code quality of AI-assisted code generation tools: An empirical study on GitHub Copilot, Amazon CodeWhisperer, and ChatGPT [Preprint]. arXiv. <https://doi.org/10.48550/arXiv.2304.10778>
 46. Zhao, S., Wang, D., Zhang, K., Luo, J., Li, Z., & Li, L. (2025). Is vibe coding safe? Benchmarking vulnerability of agent-generated code in real-world tasks [Preprint]. arXiv. <https://doi.org/10.48550/arXiv.2512.03262>
 47. Zhu, Q., Cao, J., Lu, Y., Lin, H., Han, X., Sun, L., & Cheung, S.-C. (2024). DOMAINEVAL: An auto-constructed benchmark for multi-domain code generation [Preprint]. arXiv. <https://doi.org/10.48550/arXiv.2408.13204>
 48. Ziegler, A., Kalliamvakou, E., Li, X. A., Rice, A., Rifkin, D., Simister, S., Sittampalam, G., & Aftandilian, E. (2024). Measuring GitHub Copilot's impact on productivity. *Communications of the ACM*, 67(3), 54–63. <https://doi.org/10.1145/3633453>