UDC 004.4:004.8

https://doi.org/10.52058/2786-6025-2025-10(51)-1124-1139

Kopp Andrii Mykhailovych Doctor of Philosophy, Associate Professor, Head of the Software Engineering and Management Intelligent Technologies Department, National Technical University "Kharkiv Polytechnic Institute", Kharkiv, https://orcid .org/0000-0002-3189-5623

Gamayun Igor Petrovych Doctor of Technical Sciences, Professor, Professor of the Software Engineering and Management Intelligent Technologies Department, National Technical University "Kharkiv Polytechnic Institute", Kharkiv, https://orcid .org/0000-0003-2099-4658

Nesterenko Ivan Serhiiovych Postgraduate Student of the Software Engineering and Management Intelligent Technologies Department, National Technical University "Kharkiv Polytechnic Institute", Kharkiv, https://orcid.org/ 0009-0002-5842-5426

Dashkivskyi Roman Borysovych Postgraduate Student of the Software Engineering and Management Intelligent Technologies Department, National Technical University "Kharkiv Polytechnic Institute", Kharkiv, https://orcid.org/ 0009-0006-8066-3622

DECISION-MAKING APPROACH USING LARGE LANGUAGE MODELS IN SOFTWARE DEVELOPMENT PROJECTS

Abstract. The relevance of the study is determined by the rapid development of Artificial Intelligence (AI) tools, which are increasingly used in software development projects. Choosing the best AI tools is a complex problem that requires formalized approaches to evaluating such tools. The object of the study is the process of decision-making support regarding the choice of AI tools in software development projects. The subject of the study is a method based on the use of Large Language Models (LLMs) as sources of expert judgments as part of the Saaty's pairwise comparison procedure instead of human experts. The goal of the study is to automate decision-making in software development projects based on the use of LLMs to formulate pairwise comparisons, aggregation, and evaluation of the obtained results.

The study considers the use of AI-based code generation tools such as GitHub Copilot, Amazon Q Developer, Tabnine, and Windsurf, for which pairwise

comparison matrices are created using GPT-5, Gemini-2.5-Pro, Claude-Sonnet-4, and DeepSeek-R1. The CR ≤ 0.1 coefficient confirmed the consistency of the judgments of both individual LLMs and aggregated judgments obtained based on the geometric mean method.

The calculated weights demonstrate the prevalence of GitHub Copilot (0.58) over other AI-based code generation tools. Comparison of the obtained results with analytical sources Gartner Magic Quadrant for AI Coding Assistants and Gartner Peer Insights indicates the efficiency of the approach based on the use of LLMs to automate expert judgment and increase the objectivity of decision-making when choosing AI tools for automated code generation in software development projects.

Keywords: software development, artificial intelligence, decision-making, large language model, pairwise comparison.

Копп Андрій Михайлович доктор філософії, доцент, завідувач кафедри програмної інженерії та інтелектуальних технологій управління, Національний технічний університет «Харківський політехнічний інститут», м. Харків, https://orcid.org/0000-0002-3189-5623

Гамаюн Ігор Петрович доктор технічних наук, професор, професор кафедри програмної інженерії та інтелектуальних технологій управління, Національний технічний університет «Харківський політехнічний інститут», м. Харків, https://orcid.org/0000-0003-2099-4658

Нестеренко Іван Сергійович аспірант кафедри програмної інженерії та інтелектуальних технологій управління, Національний технічний університет «Харківський політехнічний інститут», м. Харків, https://orcid.org/0009-0002-5842-5426

Дашківський Роман Борисович аспірант кафедри програмної інженерії та інтелектуальних технологій управління, Національний технічний університет «Харківський політехнічний інститут», м. Харків, https://orcid.org/0009-0006-8066-3622

ПІДХІД ДО ПРИЙНЯТТЯ РІШЕНЬ НА ОСНОВІ ВЕЛИКИХ МОВНИХ МОДЕЛЕЙ У ПРОЄКТАХ З РОЗРОБКИ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

Анотація. Актуальність дослідження визначається швидким розвитком інструментів на основі штучного інтелекту (ШІ), які все частіше застосовуються у проєктах з розробки програмного забезпечення. Вибір найкращих

ШІ-засобів є складним завданням, яке вимагає формалізованих підходів до оцінювання таких інструментів. Об'єкт дослідження – процес підтримки прийняття рішень щодо вибору ШІ-інструментів в проєктах з розробки програмного забезпечення. Предмет дослідження – метод на основі застосування великих мовних моделей (ВММ) як джерел експертних суджень у процедурі попарного порівняння Сааті замість експертів-людей. Метою дослідження є автоматизація прийняття рішень у проєктах з розробки програмного забезпечення на основі застосування ВММ для формування попарних порівнянь, агрегації та оцінювання отриманих результатів. У дослідженні розглядається використання інструментів генерації коду на основі ШІ, таких як GitHub Copilot, Amazon Q Developer, Tabnine та Windsurf, для яких матриці попарних порівнянь створені за допомогою GPT-5, Gemini-2.5-Pro, Claude-Sonnet-4 та DeepSeek-R1. Коефіцієнт CR ≤ 0,1 підтвердив узгодженість суджень як окремих ВММ, так і агрегованих суджень, отриманих на основі методу геометричного середнього. Розраховані вагові коефіцієнти демонструють перевагу GitHub Copilot (0,58) над іншими інструментами генерації коду на основі ШІ. Порівняння отриманих результатів з аналітичними джерелами Gartner Magic Quadrant for AI Coding Assistants та Gartner Peer Insights свідчить про працездатність підходу на основі використання ВММ для автоматизації експертного оцінювання та підвищення об'єктивності прийняття рішень при виборі ШІ-інструментів для автоматизованої генерації коду в проєктах з розробки програмного забезпечення.

Ключові слова: розробка програмного забезпечення, штучний інтелект, прийняття рішень, велика мовна модель, попарне порівняння.

Problem statement. Recent studies demonstrate a growing interest in the use of Artificial Intelligence tools (AI) to support decision-making in complex projects, in particular in the field of software development projects. Multi-criteria analysis methods, such as the Saaty's pairwise comparison method and the Analytic Hierarchy Process (AHP), traditionally rely on expert assessments and subjective judgments. However, the emergence of powerful language models, such as GPT-4, opened opportunities to automate the pairwise comparison process and reduce dependence on the human factor [1]. Despite this, most existing approaches are based on a single integrated agent, which limits the diversity of expert positions and does not take into account the potential of multi-agent systems for forming coordinated decisions [2].

The integration of generative AI into decision-making raises new questions about the interaction between automated assessments and traditional expert judgment. In this context, it remains important to define the role of models in the Saaty's numerical scale, where the comparison of alternatives can reflect both human-alike judgments and computational logic [3]. A comparison of the results of AHP

assessments made by humans and AI shows a difference in priorities and thinking patterns, which creates a basis for finding mechanisms to achieve consensus between different agents [4].

Previous studies also demonstrate the successful use of AI for multi-criteria evaluation in hybrid systems, where some decisions are still left to humans, while computational models play a supporting or analytical role [5]. The application of fuzzy AHP in combination with artificial intelligence algorithms confirms the potential for automation, but largely retains human control over the formation of initial judgments [6].

Individual studies demonstrate the possibility of replacing or supporting experts in pairwise comparison processes, for example, in urban planning tasks, which emphasizes the breadth of applications of this approach [7].

Thus, a relevant scientific and practical task is to develop an approach to using multiple AI assistants as sources of expert judgments within the Saaty's pairwise comparison method and, further, the AHP. The proposed approach assumes the coordination between decision-making AI assistants, increasing the objectivity of assessments, and the reproducibility of results when selecting AI tools to support software development projects.

Recent studies analysis. Recent studies indicate active development in the fields that combine the AHP with AI technologies, creating hybrid or fully automated decision support systems.

A number of studies have proposed the use of fuzzy AHP for weighting criteria in combination with neural networks for predictive assessment, demonstrating the effectiveness of integrating AI and multi-criteria decision-making methods [8].

Other approaches use AHP together with machine learning algorithms, which increases the scalability and speed of data processing in large decision-making systems [9].

Some studies have used AI assistants to collect and summarize expert judgments, modeling the pairwise comparison process in the style of human experts [10]. At the same time, the development of Python libraries with built-in AHP methods and other multi-criteria decision-making approaches confirms the potential of language models in creating automated decision-support systems [11].

The use of fuzzy AHP in combination with estimates generated by Large Language Models (LLMs) has proven effective in medical tasks where stability and interpretability of results are required [12].

Other works demonstrate the application of LLMs for forming pairwise comparisons in the context of security, highlighting the ability of AI to support risk assessment and critical indicators [13]. Practical examples of using GPT-4 to directly generate pairwise comparison matrices within AHP prove the technical feasibility of this approach [14].

An in-depth analysis of the use of multiple LLMs for parallel evaluation of alternatives revealed differences in consistency and the presence of potential biases, which is particularly important for multi-agent systems [15].

Similar experiments combine fuzzy evaluations with AHP-like weighting to evaluate the quality of chatbots, demonstrating the versatility of the method [16]. The integration of AI-assisted AHP into TOPSIS (Technique for Order of Preference by Similarity to Ideal Solution) models has opened up new opportunities for the selection of tools and technologies in various domains [17].

Additionally, the use of specially tuned LLMs has improved the stability of AHP results under conditions of uncertainty, highlighting the potential of AI in creating reliable and repeatable expert assessments [18].

In general, recent studies demonstrate a shift from hybrid to fully automated solutions, where AI not only supports but also substitutes human experts in multi-criteria decision-making processes.

This forms the basis for the development of new approaches, in particular multi-agent evaluation systems, in which several AI models can interact to achieve coordinated and informed decisions.

Research objective. The study aims to automate decision-making in software development projects based on the use of LLMs to formulate pairwise comparisons, aggregation, and evaluation of the obtained results.

The object of the study is the process of decision-making support regarding the choice of AI tools in software development projects.

The subject of the study is a method based on the use of LLMs as sources of expert judgments as part of the Saaty's pairwise comparison procedure instead of human experts.

Materials and methods. Fig. 1 demonstrates the proposed approach to using multiple AI assistants for automated pairwise comparison procedure within the AHP. The outlined procedure starts with identifying alternatives and formulating a prompt for comparison.

After that, the prompt is sent to each AI assistant. Each AI assistant forms its own comparison matrix $A^{(1)}, A^{(2)}, \ldots, A^{(m)}$, which reflects its assessment of the relative importance of the alternatives.

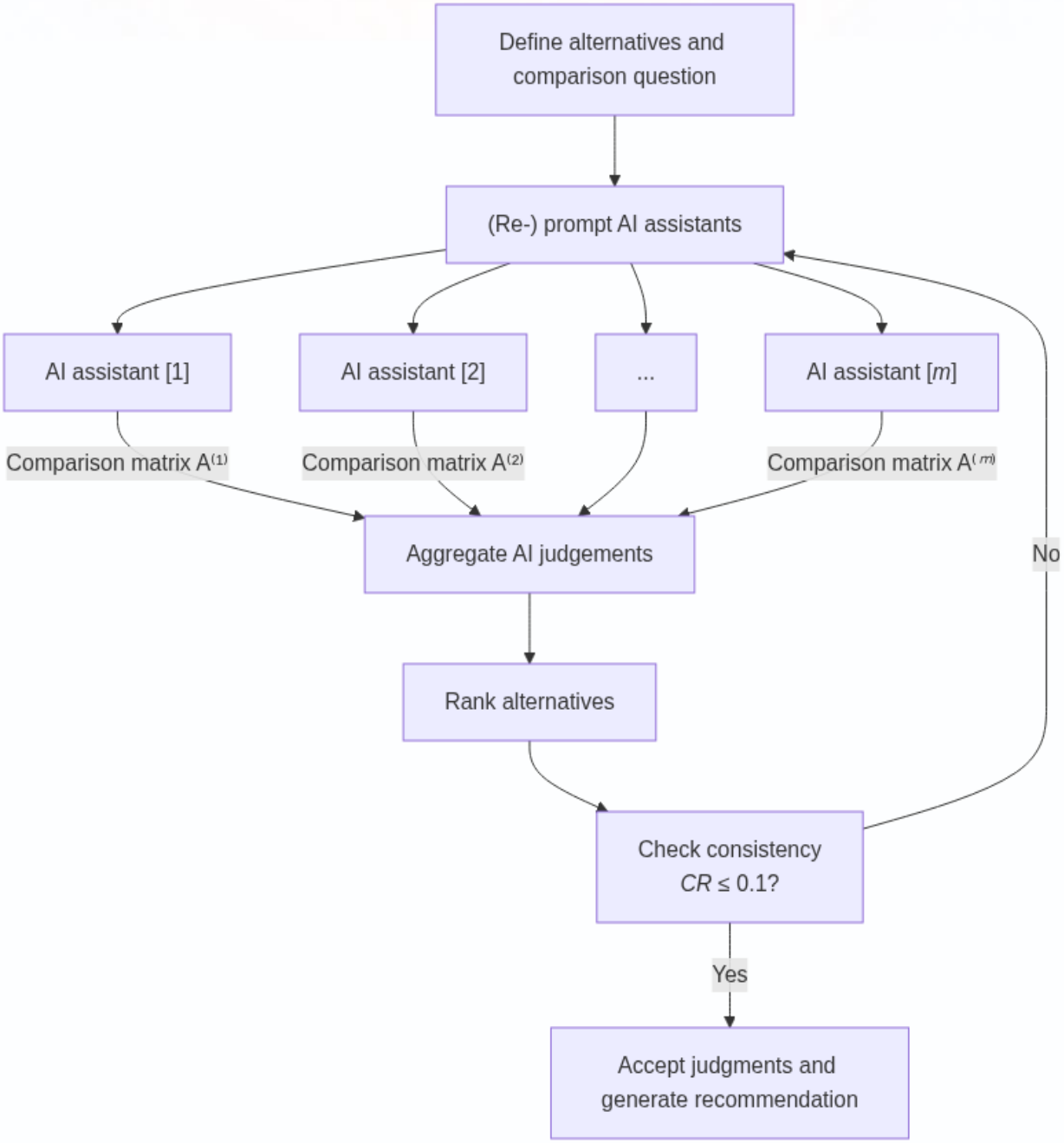

Fig. 1. Proposed approach of using AI assistants for pairwise comparison

Furthermore, the results of individual AI assistants are aggregated to obtain the generalized AI judgments. Then alternatives are ranked according to the calculated weights. Next, the consistency of judgments is checked using the $CR \leq 0.1$ rule. If the consistency condition is not met, the procedure returns to the stage of re-prompting AI assistants to refine their assessments. If the Consistency Rate (CR) is acceptable, the judgments obtained are accepted, and the system generates a final recommendation.

Table 1 outlines comparison of four modern automated code generation tools, including GitHub Copilot, Amazon Q Developer, Tabnine, and Windsurf.

Table 1

Automated AI code generation tools

Feature	GitHub Copilot	Amazon Q Developer	Tabnine	Windsurf
AI model	OpenAI Codex (GPT-3.5 or GPT-4 based)	Custom foundation KKM by AWS	Customized GPT-2 and proprietary LLMs	Proprietary LLM
Supported languages	20+ (Python, JS, TypeScript, Go, Ruby, etc.)	15+ (Python, JS, Java, TypeScript, C#, etc.)	30+ (Python, JS, Java, C++, PHP, Rust, etc.)	70+ (JS, Python, Java, C/C++, SQL, etc.)
IDE integration	VS Code, JetBrains, Neovim, Visual Studio	VS Code, JetBrains, AWS Cloud9	VS Code, JetBrains, Vim, Eclipse, etc.	VS Code, JetBrains, Vim, Emacs, Jupyter, etc.
Offline mode	No	Yes (enterprise only)	Yes (with local LLM option)	No
Context awareness	High	Moderate	Low	Moderate
Code completion quality	High	Moderate	Low	Moderate
Security scanning	Basic via GitHub advanced security	Built-in security scans (on AWS)	No	No
Pricing	$10/month (free education plan)	Free for individuals; paid for enterprise	$12/month (free basic)	Free for individuals; paid for enterprise
Privacy and data usage	Sends code to Microsoft and OpenAI servers	Anonymized data, AWS policy applies	Local option (no cloud) in pro version	Claims no telemetry, processing in RAM
Open source friendliness	Free for open source developers	Free individual use	Open source-friendly (local version available)	Free for all (as of 2025)
Unique strengths	Deep GitHub integration, state-of-the-art completion	AWS-native, strong on enterprise compliance	Best local/offline experience, fast response	Fast, lightweight, supports many languages

Table 1 shows that each AI tool has its own unique advantages depending on the development needs:

– GitHub Copilot [19], based on the OpenAI model, provides the highest quality code autocompletion and deep integration with GitHub, making it ideal for complex projects;

– Amazon Q Developer [20] provides built-in code security scanning and tight integration with the AWS environment, making it attractive to enterprise users;

– Tabnine [21] offers local operation without a cloud connection, ensuring privacy and speed, especially in the team version;

– Windsurf [22] provides free support for over 70 programming languages, multi-platform compatibility, and high generation speed, making it a convenient choice for everyday use.

All four examined AI code generation tools have different levels of contextual awareness, IDE (Integrated Development Environment) support, and pricing policies, allowing to choose the most suitable tool, assuming the specific software development project needs and development environment.

A comparative analysis of modern LLMs, including GPT-5, Gemini-2.5-Pro, Claude-Sonnet-4, and DeepSeek-R1 shows (Table 2) the high level of AI development in the field of text generation, code, and multimodal data processing.

Table 2

Most popular LLMs used as AI assistants

Feature	**GPT-5**	**Gemini-2.5-Pro**	**Claude-Sonnet-4**	**DeepSeek-R1**
Release date	August 2025	June 2025	August 2025	September 2025
Model type	Multimodal LLM	Multimodal LLM	Text LLM	Text, code LLM
Token content window	1M	2M	200K	128K
Maximum output tokens	≈ 4-8K	≈ 4-6K	≈ 8K	≈ 6K
Best use cases	Advanced research, coding, agents	Web-rich tasks, document and video understanding	Reliable writing, education, aligned reasoning	Fast coding, reasoning, light cost use cases

All of the four considered AI models differ in context window size, multimodal processing capabilities, memory usage, speed, tool integration, and security level,

allowing the model selection to be tailored to specific scientific, professional, or educational goals:

– GPT-5 [23] demonstrates the highest accuracy, logical thinking, and support for complex tools, including personal memory and code usage;

– Gemini-2.5-Pro [24] is particularly effective in tasks related to video, documents, and deep integration into the Google ecosystem;

– Claude-Sonnet-4 [25] stands out for its ethical approach, long context, and stable generation of logically consistent text, making it suitable for educational and humanitarian tasks;

– DeepSeek-R1 [26] is geared toward programmers, providing high speed, code generation accuracy, and affordability.

Fig. 2 shows the fragment of a JSON (JavaScript Object Notation) structure request used as the prompt to interact with AI assistants when forming pairwise comparisons of AI code generation tools.

```
{
  "goal": "Return this JSON with completed <FILL IN ...> fields",
  "answer-scale": "Saaty values 1/9, 1/7, 1/5, 1/3, 1, 3, 5, 7, 9",
  "explanation-type": "One sentence comment",
  "judgements": [
    {
      "is": "GitHub Copilot",
      "than": "Amazon CodeWhisperer",
      "verbal": "With respect to automated code generation, how much more
      preferred is AI tool GitHub Copilot compared to Amazon CodeWhisperer?",
      "answer": "<FILL IN SAATY SCALE VALUE>",
      "explanation": "<FILL IN TEXT EXPLANATION>"
    },
```

Fig. 2. Fragment of the JSON request sending as a prompt for AI assistants

This format ensures standardized collection of judgments and simplifies further automated aggregation of results between different AI assistants:

– “goal” field specifies instructions for the model to fill in missing values, and “answer-scale” defines the Saaty’s scale (1/9 – 9) used to evaluate the advantages of one alternative over another;

– “explanation-type” field indicates that the explanation should be short, in the form of a single sentence;

– “judgements” contains an array of comparisons, where each element defines a pair of alternatives (“is” and ‘than’) and the wording of the question in the “verbal” field.

Therefore, each AI assistant is expected to fill in the rating value in the “answer” field according to the Saaty’s scale and provide a short textual justification in the “explanation” field (Fig. 3).

```
{
  "goal": "Return this JSON with completed <FILL IN ...> fields",
  "answer-scale": "Saaty values 1/9, 1/7, 1/5, 1/3, 1, 3, 5, 7, 9",
  "explanation-type": "One sentence comment",
  "judgements": [
    {
      "is": "GitHub Copilot",
      "than": "Amazon CodeWhisperer",
      "verbal": "With respect to automated code generation, how much more
      preferred is AI tool GitHub Copilot compared to Amazon CodeWhisperer?",
      "answer": "5",
      "explanation": "Copilot generally delivers more accurate, context-aware
      multi-line code completions across languages and editors."
    },
```


Fig. 3. Fragment of the JSON response retrieved from AI assistants

The four pairwise comparison matrices were obtained using selected AI models, including GPT-5, Gemini-2.5-Pro, Claude-Sonnet-4, and DeepSeek-R1:

$$A^{(1)} = \begin{bmatrix} 1 & 5 & 7 & 3 \\ 1/5 & 1 & 3 & 1/3 \\ 1/7 & 1/3 & 1 & 1/5 \\ 1/3 & 3 & 5 & 1 \end{bmatrix}, A^{(2)} = \begin{bmatrix} 1 & 3 & 5 & 3 \\ 1/3 & 1 & 3 & 1/3 \\ 1/5 & 1/3 & 1 & 1/5 \\ 1/3 & 3 & 5 & 1 \end{bmatrix},$$

$$A^{(3)} = \begin{bmatrix} 1 & 3 & 5 & 3 \\ 1/3 & 1 & 1 & 1/3 \\ 1/5 & 1 & 1 & 1/3 \\ 1/3 & 3 & 3 & 1 \end{bmatrix}, A^{(4)} = \begin{bmatrix} 1 & 5 & 7 & 9 \\ 1/5 & 1 & 3 & 5 \\ 1/7 & 1/3 & 1 & 3 \\ 1/9 & 1/5 & 1/3 & 1 \end{bmatrix},$$

where:

– $A^{(1)}$ is the pairwise comparison matrix obtained using GPT-5;

– $A^{(2)}$ is the pairwise comparison matrix obtained using Gemini-2.5-Pro;

– $A^{(3)}$ is the pairwise comparison matrix obtained using Claude-Sonnet-4;

– $A^{(4)}$ is the pairwise comparison matrix obtained using DeepSeek-R1.

Fig. 4 demonstrates the weights of alternative AI code generation tools (Table 1), calculated based on pairwise comparison matrices generated by AI assistants.

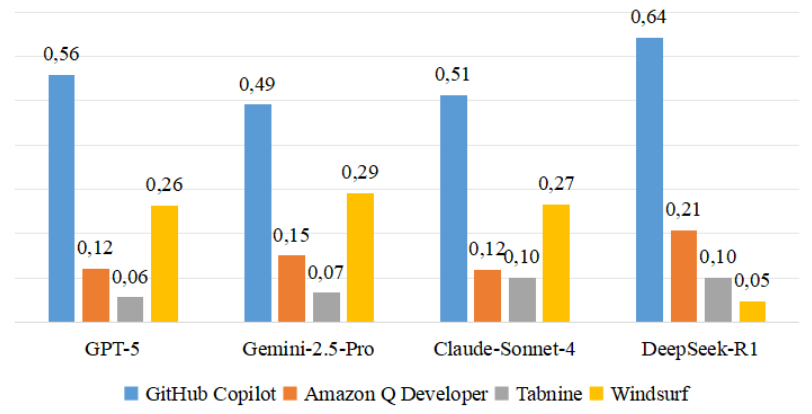

Fig. 4. Weights of AI code generation tools based on individual LLMs' judgments

The CI (Consistency Index), CR (Consistency Rate), and RI (Random Index) measures are used to assess the consistency of judgments in a pairwise comparison matrix, where CI determines the level of deviation from complete consistency, RI sets the reference random value, and CR shows the relative degree of consistency of the obtained estimates [27]:

$$CI = \frac{\lambda_{max} - n}{n - 1}, CR = \frac{CI}{RI},$$

where:

– λ_{max} is the maximum eigenvalue;

– n is the number of alternatives ($n = 4$);

– RI is the random index constant value of 0.9 for the case of 4 alternatives.

Table 3 shows the results of assessing the consistency of judgments obtained from different LLMs used for pairwise comparisons.

Table 3

Consistency assessment of the LLM-based judgments

Assessment	**GPT-5**	**Gemini-2.5-Pro**	**Claude-Sonnet-4**	**DeepSeek-R1**
λ_{max}	4.18	4.28	4.16	4.30
CI	0.06	0.09	0.05	0.10
CR	0.07	0.10	0.06	0.11
Is consistent?	Yes	Yes	Yes	Yes (with tolerated exceed)

While all models demonstrated acceptable consistency ($CR \leq 0.1$), DeepSeek-R1 showed only a slight exceedance of the acceptable level (of 0.11 vs. 0.1).

To aggregate individual pairwise comparison matrices $A^{(1)}, A^{(2)}, \ldots, A^{(m)}$ given by different AI assistants, the geometric mean was applied:

$$\bar{a}_{ij} = \left(\prod_{k=1}^{m} a_{ij}^{(k)} \right)^{1/m},$$

where:

– $a_{ij}^{(k)}$ is the element in row i, column j from expert k ($i, j = \overline{1, n}$, $k = \overline{1, m}$);

– n is the number of alternatives ($n = 4$);

– m is the number of AI assistants used as experts ($m = 4$).

This allowed obtaining a consistent aggregated judgment matrix:

$$\bar{A} = \begin{bmatrix} 1 & 3.87 & 5.92 & 3.95 \\ 0.26 & 1 & 2.28 & 0.66 \\ 0.17 & 0.44 & 1 & 0.45 \\ 0.25 & 1.52 & 2.24 & 1 \end{bmatrix}.$$

Based on the aggregated matrix of pairwise comparisons, the following weight coefficients for the alternatives were obtained (Fig. 5), with GitHub Copilot of 0.58, Windsurf of 0.18, Amazon Q Developer of 0.15, and Tabnine of 0.08, which shows GitHub Copilot's greatest advantage in the context of automated AI code generation.

The judgment consistency assessment using the aggregated pairwise comparison matrix demonstrates a high level of stability of judgments between AI assistants. The obtained maximum eigenvalue $\lambda_{max} = 4.06$ indicates a slight deviation from $n = 4$. The consistency index $CI = 0.02$ is very low, which confirms the internal logic of the formed assessments. The calculated consistency rate $CR = 0.02$ is significantly less than the threshold value of 0.1, therefore, the aggregated judgments can be considered fully consistent and reliable for further decision-making.

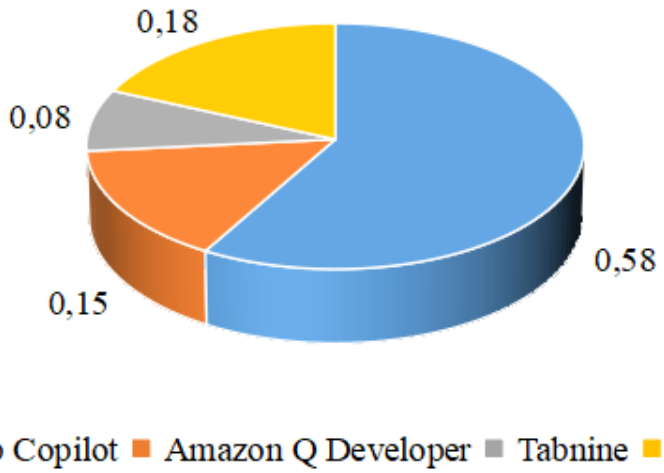

Fig. 5. Weights of AI code generation tools based on aggregated LLMs' judgments

The weighting coefficients obtained in the study are consistent with independent sources, which confirm GitHub Copilot leading position among the considered AI tools for automated code generation:

– Gartner Magic Quadrant for AI Coding Assistants [28], where GitHub Copilot is positioned in the Leaders quadrant, closest to the top right corner, indicating its strategic maturity and highest level of completeness of vision among competitors;

– Gartner Peer Insights platform [29], where GitHub Copilot ranked first among the tools reviewed with an average rating of 4.4 (out of 5) based on 190 reviews.

Conclusions. In this study, the following results were obtained:

1. The study proposed a new multi-assistant AI-based approach to the automation of the Saaty's pairwise comparison process, in which several LLMs are used as sources of expert judgments. This made it possible to replace traditional human assessments with consistent AI judgments and ensure the reproducibility of decision-making.

2. The results demonstrated that all pairwise comparison matrices generated by LLMs have acceptable consistency ($CR \leq 0.1$), and the aggregate matrix obtained using the geometric mean demonstrates high stability of judgments ($\lambda_{max} = 4.06$, $CI = 0.02$, $CR = 0.02$), which confirms the logicality and reliability of the obtained estimates.

3. Based on the aggregated judgments, the weight coefficients of the alternatives were determined, among which GitHub Copilot received the highest weight (0.58), ahead of Windsurf (0.18), Amazon Q Developer (0.15), and Tabnine (0.08) which indicates its leading position among automated code generation tools.

4. The obtained results correlate with independent analytical sources, in particular the Gartner Magic Quadrant for AI Coding Assistants, where GitHub Copilot is placed in the leaders quadrant, as well as with Gartner Peer Insights assessments, where it took first place with an average rating of 4.4 out of 5, which confirms the reliability of the proposed AI-based approach to decision-making support in the field of software development project.

In the future work, it is planned to expand the proposed approach by combining multi-criteria analysis using the AHP with LLM-based judgments for evaluating the effectiveness of AI coding tools in software development projects.

References:

1. Svoboda, I., & Lande, D. (2024). Enhancing Multi-Criteria Decision Analysis with AI: Integrating Analytic Hierarchy Process and GPT-4 for Automated Decision Support. – Retrieved from https://arxiv.org/pdf/2402.07404

2. Svoboda, I., & Lande, D. (2024). Ai Agents in Multi-Criteria Decision Analysis: Automating the Analytic Hierarchy Process with Large Language Models. – Retrieved from https://doi.org/10.2139/ssrn.5069656

3. Agarwal, A. (2025). Optimizing employee roles in the era of generative AI: a multi-criteria decision-making analysis of co-creation dynamics. – Retrieved from https://doi.org/10.1080/23311886.2025.2476737

4. Srđević, B. (2025). Evaluating the Societal Impact of AI: A Comparative Analysis of Human and AI Platforms Using the Analytic Hierarchy Process. – Retrieved from https://doi.org/10.3390/ai6040086

5. Radulescu, C. Z., & Radulescu, M. (2025). Criteria Analysis for the Selection of a Generative Artificial Intelligence Tool for Academic Research Based on an Improved Group DEMATEL Method. – Retrieved from https://doi.org/10.3390/app15105416

6. Gupta, S., Kaur, S., Gupta, M., & Singh, T. (2024). AI empowered academia: a fuzzy prioritization framework for academic challenges. Journal of International Education in Business. – Retrieved from https://doi.org/10.1108/jieb-06-2024-0071

7. Sofia, D. (2025). Redefining Choices. How Does Artificial Intelligence Support Decision-Making in Urban and Architectural Development? – Retrieved from https://webthesis.biblio.polito.it/secure/34480/1/tesi.pdf

8. Adepoju, S.A. (2019). Integrated usability evaluation framework for university websites. – Retrieved from https://doi.org/10.26634/jit.8.1.15713

9. Alves, M. A., Meneghini, I. R., Gaspar-Cunha, A., & Guimarães F. G. (2023). Machine Learning-Driven Approach for Large Scale Decision Making with the Analytic Hierarchy Process. – Retrieved from https://doi.org/10.3390/math11030627

10. Lee, A. H. I., & Kang, H.-Y. (2023). A Three-Phased Fuzzy Logic Multi-Criteria Decision-Making Model for Evaluating Operation Systems for Smart TVs. – Retrieved from https://doi.org/10.3390/app13137869

11. Pereira, V., Basilio, M. P., & Carlos, S. (2024). Enhancing Decision Analysis with a Large Language Model: pyDecision a Comprehensive Library of MCDA Methods in Python. – Retrieved from https://arxiv.org/abs/2404.06370

12. An In-Depth Analysis of the Adoption of Large Language Models in Clinical Settings: A Fuzzy Multi-Criteria Decision-Making Approach. – Retrieved from https://search.proquest.com/openview/afe33dbe92c45414a67a4e7b377f2716

13. Cheng, Y., Zhang, Q., & Zhang, X. (2024). Exploring the Integration of Large Language Models and Analytical Hierarchy Process (AHP) for Multi-Indicator Importance Analysis in the Safety Research. – Retrieved from https://doi.org/10.1109/acait63902.2024.11022119

14. Dehghanimohammadabadi, M., & Nihan Kabadayı. (2024). The Ai-Driven Decision-Making (Aidm) Framework: Integrating Ahp and Chatgpt-4 for Supplier Selection. – Retrieved from https://doi.org/10.2139/ssrn.4997750

15. Stelmach, A. (2025). Evaluating AI Judgements: A Case Study of LLMs in Product Development. – Retrieved from https://www.diva-portal.org/smash/get/diva2:1965527/FULLTEXT01.pdf

16. Ilieva, G. (2025). Extension of Interval-Valued Hesitant Fermatean Fuzzy TOPSIS for Evaluating and Benchmarking of Generative AI Chatbots. – Retrieved from https://doi.org/10.20944/preprints202501.0499.v1

17. Lachgar, M., Hrimech, H., Ommane, Y., & Laannaoui, M. D. (2024). Holistic approach for selecting chatbot development tools: combining AHP and TOPSIS methodologies. – Retrieved from https://doi.org/10.1080/2573234x.2024.2399087

18. Park, H., Oh, H., Gao, F., & Kwon, O. (2025). Enhancing Analytic Hierarchy Process Modelling Under Uncertainty With Fine-Tuning LLM. – Retrieved from https://doi.org/10.1111/exsy.70051

19. GitHub Copilot. – Retrieved from https://github.com/features/copilot

20. Amazon Q Developer. – Retrieved from https://aws.amazon.com/q/developer/

21. Tabnine AI Code Assistant. – Retrieved from https://www.tabnine.com/

22. Windsurf. – Retrieved from https://windsurf.com/.

23. Introducing GPT-5. – Retrieved from https://openai.com/index/introducing-gpt-5/

24. Gemini 2.5 Pro. – Retrieved from https://deepmind.google/models/gemini/pro/
25. Claude Sonnet 4.5. – Retrieved from https://www.anthropic.com/claude/sonnet
26. DeepSeek-R1. – Retrieved from https://huggingface.co/deepseek-ai/DeepSeek-R1
27. Vargas, R. V. (2019). Using the analytic hierarchy process (AHP) to select and prioritize projects in a portfolio. – Retrieved from https://www.pmi.org/learning/library/analytic-hierarchy-process-prioritize-projects-6608
28. Qodo Named a Visionary in the September 2025 Gartner Magic Quadrant for AI Coding Assistants. – Retrieved from https://www.qodo.ai/reports/gartner-report-ai-code-assistants-2025/
29. Best AI Code Assistants Reviews 2024. – Retrieved from https://www.gartner.com/reviews/market/ai-code-assistants

Література:

1. Svoboda, I., & Lande, D. (2024). Enhancing Multi-Criteria Decision Analysis with AI: Integrating Analytic Hierarchy Process and GPT-4 for Automated Decision Support [Електронний ресурс]. – Режим доступу: https://arxiv.org/pdf/2402.07404
2. Svoboda, I., & Lande, D. (2024). Ai Agents in Multi-Criteria Decision Analysis: Automating the Analytic Hierarchy Process with Large Language Models [Електронний ресурс]. – Режим доступу: https://doi.org/10.2139/ssrn.5069656
3. Agarwal, A. (2025). Optimizing employee roles in the era of generative AI: a multi-criteria decision-making analysis of co-creation dynamics [Електронний ресурс]. – Режим доступу: https://doi.org/10.1080/23311886.2025.2476737
4. Srđević, B. (2025). Evaluating the Societal Impact of AI: A Comparative Analysis of Human and AI Platforms Using the Analytic Hierarchy Process [Електронний ресурс]. – Режим доступу: https://doi.org/10.3390/ai6040086
5. Radulescu, C. Z., & Radulescu, M. (2025). Criteria Analysis for the Selection of a Generative Artificial Intelligence Tool for Academic Research Based on an Improved Group DEMATEL Method [Електронний ресурс]. – Режим доступу: https://doi.org/10.3390/app 15105416
6. Gupta, S., Kaur, S., Gupta, M., & Singh, T. (2024). AI empowered academia: a fuzzy prioritization framework for academic challenges. Journal of International Education in Business [Електронний ресурс]. – Режим доступу: https://doi.org/10.1108/jieb-06-2024-0071
7. Sofia, D. (2025). Redefining Choices. How Does Artificial Intelligence Support Decision-Making in Urban and Architectural Development? – Retrieved from https://webthesis.biblio.polito.it/secure/34480/1/tesi.pdf
8. Adepoju, S.A. (2019). Integrated usability evaluation framework for university websites [Електронний ресурс]. – Режим доступу: https://doi.org/10.26634/jit.8.1.15713
9. Alves, M. A., Meneghini, I. R., Gaspar-Cunha, A., & Guimarães F. G. (2023). Machine Learning-Driven Approach for Large Scale Decision Making with the Analytic Hierarchy Process [Електронний ресурс]. – Режим доступу: https://doi.org/10.3390/math11030627
10. Lee, A. H. I., & Kang, H.-Y. (2023). A Three-Phased Fuzzy Logic Multi-Criteria Decision-Making Model for Evaluating Operation Systems for Smart TVs [Електронний ресурс]. – Режим доступу: https://doi.org/10.3390/app13137869
11. Pereira, V., Basilio, M. P., & Carlos, S. (2024). Enhancing Decision Analysis with a Large Language Model: pyDecision a Comprehensive Library of MCDA Methods in Python [Електронний ресурс]. – Режим доступу: https://arxiv.org/abs/2404.06370
12. An In-Depth Analysis of the Adoption of Large Language Models in Clinical Settings: A Fuzzy Multi-Criteria Decision-Making Approach [Електронний ресурс]. – Режим доступу: https://search.proquest.com/openview/afe33dbe92c45414a67a4e7b377f2716

13. Cheng, Y., Zhang, Q., & Zhang, X. (2024). Exploring the Integration of Large Language Models and Analytical Hierarchy Process (AHP) for Multi-Indicator Importance Analysis in the Safety Research [Електронний ресурс]. – Режим доступу: https://doi.org/10.1109/acait63902.2024.11022119

14. Dehghanimohammadabadi, M., & Nihan Kabadayı. (2024). The Ai-Driven Decision-Making (Aidm) Framework: Integrating Ahp and Chatgpt-4 for Supplier Selection [Електронний ресурс]. – Режим доступу: https://doi.org/10.2139/ssrn.4997750

15. Stelmach, A. (2025). Evaluating AI Judgements: A Case Study of LLMs in Product Development [Електронний ресурс]. – Режим доступу: https://www.diva-portal.org/smash/get/diva2:1965527/FULLTEXT01.pdf

16. Ilieva, G. (2025). Extension of Interval-Valued Hesitant Fermatean Fuzzy TOPSIS for Evaluating and Benchmarking of Generative AI Chatbots [Електронний ресурс]. – Режим доступу: https://doi.org/10.20944/preprints202501.0499.v1

17. Lachgar, M., Hrimech, H., Ommane, Y., & Laannaoui, M. D. (2024). Holistic approach for selecting chatbot development tools: combining AHP and TOPSIS methodologies [Електронний ресурс]. – Режим доступу: https://doi.org/10.1080/2573234x.2024.2399087

18. Park, H., Oh, H., Gao, F., & Kwon, O. (2025). Enhancing Analytic Hierarchy Process Modelling Under Uncertainty With Fine-Tuning LLM [Електронний ресурс]. – Режим доступу: https://doi.org/10.1111/exsy.70051

19. GitHub Copilot [Електронний ресурс]. – Режим доступу: https://github.com/features/copilot

20. Amazon Q Developer [Електронний ресурс]. – Режим доступу: https://aws.amazon.com/q/developer/

21. Tabnine AI Code Assistant [Електронний ресурс]. – Режим доступу: https://www.tabnine.com/

22. Windsurf [Електронний ресурс]. – Режим доступу: https://windsurf.com/.

23. Introducing GPT-5 [Електронний ресурс]. – Режим доступу: https://openai.com/index/introducing-gpt-5/

24. Gemini 2.5 Pro [Електронний ресурс]. – Режим доступу: https://deepmind.google/models/gemini/pro/

25. Claude Sonnet 4.5 [Електронний ресурс]. – Режим доступу: https://www.anthropic.com/claude/sonnet

26. deepseek-ai/DeepSeek-R1 [Електронний ресурс]. – Режим доступу: https://huggingface.co/deepseek-ai/DeepSeek-R1

27. Vargas, R. V. (2019). Using the analytic hierarchy process (AHP) to select and prioritize projects in a portfolio [Електронний ресурс]. – Режим доступу: https://www.pmi.org/learning/library/analytic-hierarchy-process-prioritize-projects-6608

28. Qodo Named a Visionary in the September 2025 Gartner Magic Quadrant for AI Coding Assistants [Електронний ресурс]. – Режим доступу: https://www.qodo.ai/reports/gartner-report-ai-code-assistants-2025/

29. Best AI Code Assistants Reviews 2024 [Електронний ресурс]. – Режим доступу: https://www.gartner.com/reviews/market/ai-code-assistants