References

[1] Rajpurkar P., Irvin J., Zhu K., et al., CheXNet: Radiologist-Level Pneumonia Detection on Chest X-Rays with Deep Learning, 2017. https://doi.org/10.48550/arXiv.1711.05225

[2] Esteva A., Robicquet A., Ramsundar B., et al., A guide to deep learning in healthcare. Nature Medicine, 2019, 25(1), 24–29. DOI: 10.1038/s41591-018-0316-z

[3] Litjens G., Kooi T., Bejnordi B. E., et al. A survey on deep learning in medical image analysis. Medical Image Analysis, 2017, 42, 60–88. DOI: 10.1016/j.media.2017.07.005

[4] Kermany D.S., Zhang K., Goldbaum M., Identifying Medical Diagnoses and Treatable Diseases by Image-Based Deep Learning, 2018, Cell, 172(5), 1122–1131.e9. DOI: 10.1016/j.cell.2018.02.010

ARTIFICIAL INTELLIGENCE FOR SOFTWARE DEVELOPMENT WITH ECONOMIC SECURITY IN MIND

Nurmashova M.O.[1]

[1]Karaganda Buketov University, Karaganda, Kazakhstan
[1]E-mail: mnurmashova98@mail.ru

In the modern world, where technologies are rapidly developing and information flows are increasing exponentially, software is becoming a critical component in the functioning of both private businesses and government agencies. At the same time, the growth in complexity and scale of software systems inevitably entails risks not only of a technical but also of an economic nature. Errors in the code, vulnerabilities in the architecture, inefficient allocation of resources - all this can lead to significant financial losses, and in some cases even to the undermining of trust in the company or government body.

Modern trends in digital transformation require not only accelerated development of software products, but also ensuring their resilience to economic threats: losses from cyberattacks, inefficient use of resources, costs of fixing errors. One of the promising tools for solving these problems is the use of artificial intelligence at all stages of the software life cycle [1].

Economic software security in this context means minimizing the financial risks associated with the development, implementation and operation of IT systems. The emergence and rapid development of artificial intelligence (AI) technologies offers new tools for solving these problems. AI can not only automate routine operations, but also provide intelligent analysis of source code, predict possible failures and optimize business logic from a cost perspective.

Economic security in information systems is not only protection against squirrel web, but also a broader concept that includes the sustainability of business processes, predictability of costs and reduction of the probability of losses due to technological errors. From the point of view of software engineering, the following key aspects of economic security can be distinguished:

- Optimization of software life cycle costs (development, testing, implementation, maintenance);
- Resilience to failures, attacks, and invalid data;
- Minimizing technology debt and controlling architecture quality;
- Increase development efficiency through automation and intelligence.

Ensuring these areas requires a comprehensive approach, where AI becomes not just an auxiliary tool, but a strategic element of the digital security architecture. [2].

The development of language models (LLM), neural network systems, and deep learning algorithms has opened up new horizons in software development. The most notable areas of application of AI are:

Code generation and autocompletion

Modern systems such as GitHub Copilot, Amazon CodeWhisperer, and OpenAI Codex can offer developers ready-made code snippets based on the context of the current task. This reduces the time spent writing routine elements, promotes unification of style, and, with proper testing, reduces the likelihood of logical errors. According to available data, the use of AI tools in a number of projects has reduced development time by up to 30

Vulnerability analysis and risk identification

AI-based code analysis systems (such as DeepCode or CodeQL) can find hidden vulnerabilities that are not visible to traditional static analyzers. Furthermore, as part of the DevSecOps approach, they are integrated into the CI/CD pipeline, ensuring early detection of potential threats, both technical and financial (such as unauthorized access to payment modules).

Predictive analytics and cost control

AI is used to analyze logs, change history, user behavior, and other metrics. Such data allows predicting which software components are prone to failures, where costs may increase, and modeling project development scenarios from a budgeting and economic sustainability perspective.

Intelligence testing

AI testers are able to automatically generate test scenarios, find boundary conditions, detect anomalies in input data, and simulate system behavior under stress. This not only improves software quality, but also reduces the cost of manual testing and bug fixing at a later stage [6].

Despite the high efficiency of AI tools, a number of limitations should be considered:

- Opacity of AI decisions. Most AI models (especially neural networks) are “black boxes,” making it difficult to verify and interpret decisions in critical systems;
- Dependence on training data. The quality of AI decisions directly depends on the data on which it was trained. Incorrect or irrelevant data can lead to incorrect recommendations;
- -Legal aspects. Issues of copyright on generated AI code, as well as legal liability for errors made by AI, remain unresolved at the legislative level in most countries;
- Ethical issues. Development automation can reduce the need for a number of IT specialists, which raises debate about the social responsibility of implementing such solutions [5].

Measures to ensure economic security:

- Cybersecurity systems focused on protecting AI solutions.
- Risk assessment when implementing AI, including auditing the models and data used.

The use of artificial intelligence in software development opens up fundamentally new opportunities both in terms of technological efficiency and economic security. Cost reduction, reduced probability of critical errors, risk prediction and automation of complex processes - all this becomes a reality thanks to the implementation of AI.

However, to realize the potential of AI, it is necessary to take into account existing limitations, develop a regulatory framework, and, most importantly, train a new generation of specialists who possess not only technical skills, but also systems thinking. The development of AI in software engineering is not just a stage of evolution, but a paradigm shift that requires rethinking processes, roles, and priorities in the digital economy.

References

[1] Soloviev V.D., Chernov S.V. Artificial intelligence and intelligent systems. – M.: Hotline – Telecom, 2020. – 368 p.

[2] Bogatyrev Yu.V., Matveev A.V. Economic security in the digital economy. – St. Petersburg: Piter, 2021. – 304 p.

[3] Chen M., Tworek J., Jun H., et al. Evaluating large language models trained on code // arXiv preprint arXiv:2107.03374. – 2021.

[4] Wang W., Lee R., Wang C., et al. Software vulnerability detection using deep learning: A survey // IEEE Access. – 2021. – Vol. 9. – P. 10486-10504.

[5] Harman M., Mansouri S.A., Zhang Y. Search-based software engineering: Trends, techniques and applications// ACM Computing Surveys. – 2012. – Vol. 45(1).

[6] Panichella S., Oliveto R., Di Penta M., et al. How developers use static analysis tools in their daily work // IEEE Int. Conf. on Program Comprehension, 2015. – P. 1–10.

[7] Mittelstadt B., Russell C., Wachter S. Explaining explanations in AI// Communications of the ACM. – 2019. – Vol. 61(5). – P. 56-65.

ON THE QUESTION OF THE APPLICABILITY OF MODEL THEORY TO THE CHOICE OF THE BEST CLASS OF MACHINE LEARNING METHOD

Popova N.[1], **Issayeva A.**[2], **Samoylova I.**[3], **Pankin Y.**[4]

[1,2,3]Karaganda Buketov University, Karaganda, Kazakhstan
[4]Kazaktelekom, Karaganda, Kazakhstan
[1]E-mail: popovanv092024@gmail.com
[2]E-mail: isaevaaiga@gmail.com
[3]E-mail: irinasam2005@mail.ru

The choice of a suitable machine learning method is traditionally based on empirical methods such as cross–validation, bootstrap, and heuristics. However, with the rapidly increasing complexity of models and the variety of tasks, there is a need for theoretically sound approaches. One of the potential directions is the use of concepts from the theory of models of first-order logic, which allows us to formalize the behavior of learning algorithms through the properties of the classes of functions described by them [1].

The concept of VC dimension (Vapnik–Chervonenkis dimension) characterizes the ability of a class of functions to implement all possible partitions of a finite set of points into positive and negative examples. A class of functions is considered learnable if its VC dimension is finite [2]. This means that there is a limit to its complexity, providing a controlled generalizing ability.

The VC-dimension is actively used in establishing upper bounds on generalization error within the framework of PAC learning. For example, linear classifiers in $\mathrm{R}^n have a VC-dimension of n+1$, which makes them learnable and predictable when the number of features is limited.

The Non-Independence Property (NIP) describes the behavior of first-order theories in which it is impossible to construct formulas capable of independently encoding infinite data sets. From a formal perspective, NIP excludes the possibility of shattering infinite sets and is associated with