Using the Internet of Everything for Machine-Learning-Based Computer System Design and Optimization

Shaojun Feng, Jiaqing Zhong, and Juan Chen(✉)

College of Computer Science and Technology, National University of Defense Technology, Hunan 410073, China
{fengshaojun,zhongjiaqing23,juanchen}@nudt.edu.cn

Abstract. Machine learning (ML) has been widely used in computer system development and optimization levels, boosting computer design and optimization improvement. With the increase of computer system design complexity and the boost of the demand for software and application optimization, the difficulties of computer system problem solving are increasing, stimulating the development of ML methods. ML models represent computer system design and development approaches; combining these with the Internet of Everything (IoE) cannot be ignored. For ML models, all kinds of computer system sample data are the fundamental prerequisite for quantifying the computer system characteristics; the connection (internet) of all computer devices is critical for predicting computer system behaviors. The IoE enables more computer devices and ML models to be connected, extending how computer systems can be well developed. The IoE will affect the applications between computer system development and various ML models more deeply by connecting people, data, and machines. This work provides an overview of the history of the IoE and ML-based computer system development. It shows the interaction between ML models and computer system design and optimization. It also emphasizes the people-to-people (P2P), people-to-machine (P2M), and machine-to-machine (M2M) in the ML-based computer system design and optimization. It discusses the interpretability of ML-based computer system development.

Keywords: Internet of Everything · machine learning · computer system · people to people (P2P) · people to machine (P2M) · machine to machine (M2M)

1 Introduction

With the rapid development of digitalization, humans are constantly innovating digital technologies to connect the physical world, and the Internet of Everything (IoE) has gradually become a new, significant, and helpful concept. The establishment of the Internet of Everything is based on the Internet of Things (IoT) [1], which is different from the Internet of Things that connects physical devices. Its connectivity is more

T. Pereira et al. (Eds.): IoECon 2024, LNICST 607, pp. 67–78, 2025.
https://doi.org/10.1007/978-3-031-84426-3_6

comprehensive than the Internet of Things. The Internet of Everything is a vast network that combines people, data, and things, and components interact and exchange data in real-time, an extension and expansion of the Internet of Things. With the concept of interconnection of everything, almost everything can be connected through the internet.

In the process of computer system development, machine learning, as an interdisciplinary field, has been widely applied to different levels of computer system development and optimization, significantly promoting improving computer design and optimization levels. The Internet of Everything provides excellent assistance in developing and optimizing computer systems based on machine learning (ML). For machine learning models, various sample data of computer systems are the fundamental prerequisite for quantifying the characteristics of computer systems. The connection of all computer devices (the internet) is crucial for predicting the behavior of computer systems, and ML models provide optimization methods by learning the data features in the computer system. At the same time, with the increasing complexity of computer system design and the growing demand for software and application optimization, the difficulty of solving computer system problems is also constantly increasing, stimulating the development of machine learning algorithms.

This article briefly overviews the history and relationship between the Internet of Everything and ML-based computer system development. It introduces the development background of the Internet of Everything and ML-based computer system development, outlines the interaction between ML models and computer system design and optimization, emphasizes P2P, P2M, and M2M in the ML-based computer system design and optimization process, and predicts future development. Finally, this article discusses the interpretability issues of ML-based computer system development.

2 Background of IoE and ML-Based Computer System Development

2.1 Background of IoE

The Internet of Everything (IoE) is built on the foundation of the internet and the Internet of Things (IoT). The internet originated in the 1960s with the development of the military interconnection network ARPANET by the US Department of Defence [2]. With the continuous development of information technology and communication technology, the internet has seen explosive growth and continues to drive the digitization of the entire world. 2005 saw the emergence of the Internet of Things [3], a broader and more complex network of internet-connected devices that sense and share data and securely interact based on internet protocols. The Internet of Everything, as shown in Fig. 1, encompasses people, things, and data, and connecting them will profoundly affect how humans behave.

2.2 Background of Machine Learning

Machine learning has existed for decades or arguably centuries [4]. Back in the 17th century, Bayes, Laplace's derivation of least squares, and Markov chains were the tools and foundations of machine learning in widespread use [5].

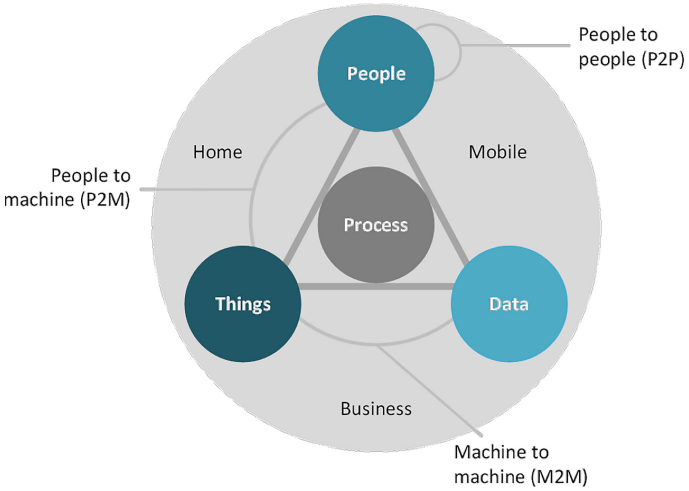


Fig. 1. The components of the Internet of Everything (IoE) [1]

The specific history of machine learning can be traced back to the 1950s when the introduction of the Turing test and the development of the checker's program by Arthur Samuel marked the formal entry of machine learning into the developmental period. However, between the mid-1960s and the end of the 1970s, the development of machine learning was almost stagnant [4, 5]. It was in the 1980s that the idea of multilayer perceptron (MLP) was introduced, using the neural network backpropagation algorithm [6], which brought machine learning into a renaissance period. In the 1990s, the proposed AdaBoost [7] and later support vector machine (SVM) [8] algorithms shifted machine learning from a knowledge-driven to a data-driven mindset. At the beginning of the 21st century, Hinton et al. [9] proposed the concept of deep learning, which made machine learning research enter a booming period again. Starting in 2012, with the improvement of computational power and the support of massive training samples, deep learning has become a hotspot of machine learning research. It has led to a wide range of applications in the industry. Machine learning can be divided into three categories: supervised learning, unsupervised learning, and reinforcement learning. Figures 2, 3 and 4 briefly list some algorithms of these three categories of machine learning.

2.3 Background of ML-Based Computer System Development

The history of hardware design optimization using machine learning can be traced back to 1977. Traditional electronic design often requires much time to determine critical parameters during the design process and does not predict what will happen after actual production. Mockus [29] then used Bayesian optimization to electronic design automation (EDA) algorithms for automatic tuning and achieved surprising results.

With further advances in machine learning, trained ML models can help speed up tape-out times by predicting downstream outcomes in the chip design flow, thus reducing the time per iteration or improving the quality of results (QoR), such as performance,

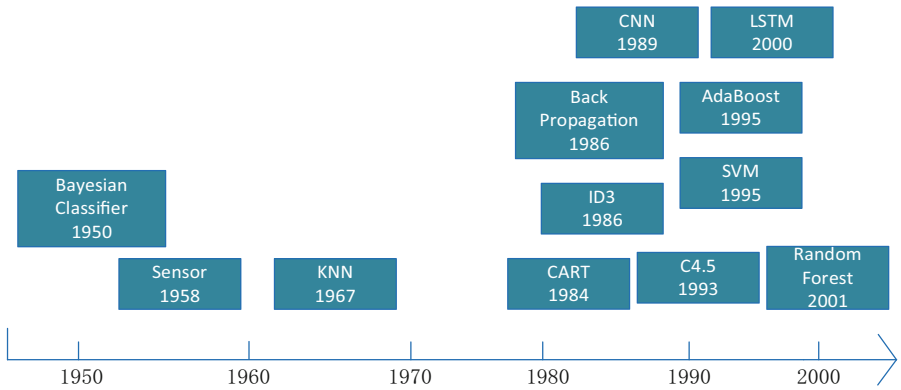
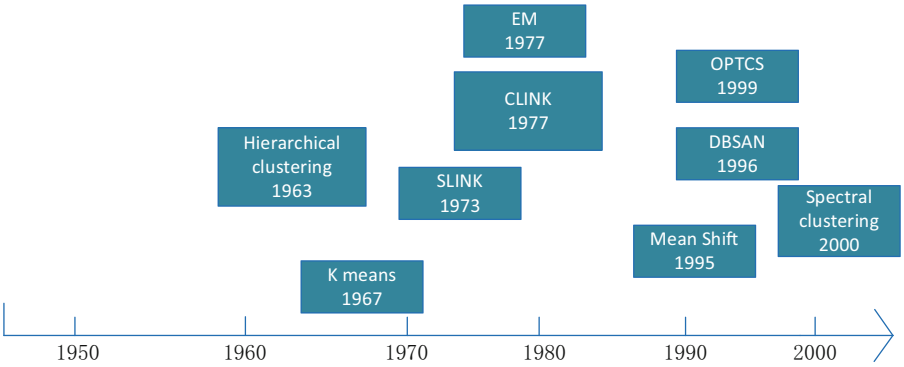
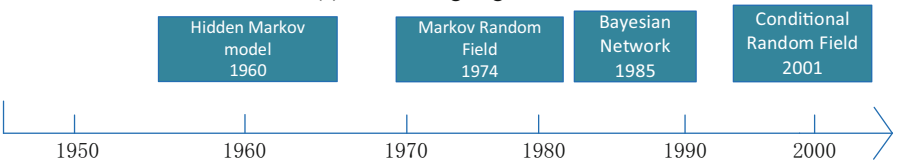


Fig. 2. Supervised Learning [7–13]



(a) Clustering Algorithms



(b) Probability Graph Models

Fig. 3. Unsupervised Learning [14–20]

power, or area. Predictions can be provided in seconds rather than waiting hours or days for accurate results. One example is ParaGraph [30], a GNN model that directly predicts layout parasitics and device parameters from circuit schematics. Post-layout parasitics prediction is vital for automated analog layout generation as it can help with schematic and layout convergence, floorplan feasibility, or QoR estimation.

In addition to deep learning algorithms, chip designers can train models to predict circuit layout signal interference and thermal distribution issues. These predictions can be used to optimize the chip layout during the design phase, dramatically improving

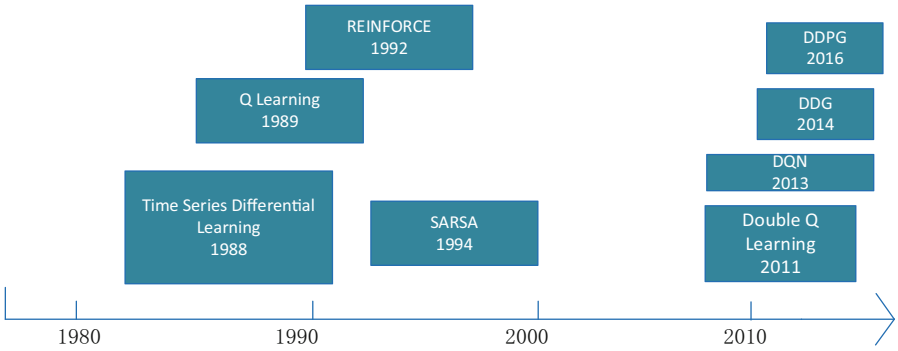


Fig. 4. Reinforcement Learning [21–28]

the efficiency and accuracy of the design. Practical examples [31] show that chips optimized using deep learning can reduce power consumption by up to 20% and improve performance by up to 30% compared to traditional methods.

When evaluating intelligent routing strategies and their effectiveness, machine learning algorithms can improve routing speed, optimize signal integrity, and reduce power consumption. For example, a high-performance chip with an artificial intelligence (AI) routing strategy [32] has a maximum transmission speed increase of about 15% over the traditional method while ensuring signal integrity. The routing also has an increased wiring density, which saves cost and board space.

The history of using machine learning to optimize system software can be traced back to 1998 when Warren Smith et al. [33] proposed a technique for predicting parallel application runtime and thus optimizing operating system scheduling by determining application similarity through unsupervised learning. In 2005, Atul Nwgi et al. [34] used the Waikato Knowledge Analysis Environment (Weka), an open-source machine learning tool, used a decision tree algorithm to learn the CPU time slice utilization behavior of known programs in Linux systems for predictive scheduling. The results showed that the process turnaround time (TaT) was reduced by 1.4% to 5.8%. However, because machine learning technology still needs to be mature enough and arithmetic power is insufficient, the above attempts to use ML techniques to generate or improve OS policies have not been adopted on a production scale.

Into the 10–20 s of this century, with the development of arithmetic power, as well as the emergence of more and more excellent models in machine learning, more developers in the field of system software faced with the increase in the complexity of system software design problems began to pay attention to: whether machine learning can be used as the basis for the processing and optimization of the problem scenarios faced in the development of system software. Ma et al. [35] then perceived that combining machine learning with databases is possible, using a supervised learning KNN clustering algorithm to classify the data by labels and LSTM to predict future workload patterns with good performance results.

In addition, machine learning (ML) techniques are emerging in OS design to build and configure OSs and "learn" OS features. For example, Zhang et al. [36] use ML to predict the optimal configuration of an OS, and such configurations can be adapted to

application changes at runtime. ML can also generate policies and mechanisms for certain OS features. The considerable engineering effort of OS development can be avoided by designing and building frameworks to train and use ML models for OSes. In addition, ML-based approaches may produce better results that are suited to different applications. Many data structures in storage systems are similar, such as B+ tree, LSM-tree, LSH-table, etc. Still, they have different application scenarios (e.g., it is more appropriate to use LSM than B+ Tree in the KV Store), which suggests that there is a space for them to replace each other. This can be done by using machine learning algorithms to select different data structures to optimize the storage system, automatically selecting a suitable underlying data structure based on the specific workload and hardware [37].

At the application level, deep integration of ML into application design can leverage more significant advantages over traditional applications. For example, the concept of a self-driving database, unlike previous work on optimizing databases with ML, a self-driving DB focuses more on how to deeply integrate ML and DB rather than engaging in one plug-in module after another [38]. In another example, deepmind feeds a large amount of relevant information collected by Google's data centers into a machine learning model that learns independently and ultimately determines which cooling configuration reduces energy consumption. The Google Data Center [39] says the project will save millions of dollars in energy consumption and could help the company reduce its carbon footprint.

3 Impact of the IoE on ML-Based Computer System Development

As shown in Fig. 5, in the research of computer system development based on machine learning, the Internet of Everything connects people, data, and things. In this research process, the people-to-people, machine-to-machine, and people-to-machine systems are evident.

In the development of machine learning-based computer systems, people-to-people systems include three representative interactions: the interaction between hardware developers and system software developers (H2S/S2H), the interaction between system software developers and application software developers (S2A/A2S), and the interaction between machine learning researchers and computer system developers (M2H/H2M, M2S\S2M, M2A\A2M). Hardware designers provide the hardware environment and user manuals for system software development. System software developers fully explore the hardware features and provide feedback on optimization suggestions for the next generation of hardware. System software researchers provide the system environment and user manuals for application development. Application developers fully explore system software features and provide feedback on optimization suggestions for the next generation of system software. Machine learning researchers guide developers in all areas of computer systems (hardware, system software, applications), using machine learning models to address or optimize complex scenarios faced in developing computer systems. Scholars in machine learning have, in turn, developed machine learning algorithmic models that are closer to real-world scenarios by studying problems in the systems domain.

The machine-to-machine system consists of two parts, as shown in the bottom section of Fig. 5: The quantifiable feature data generated by computer systems at runtime is

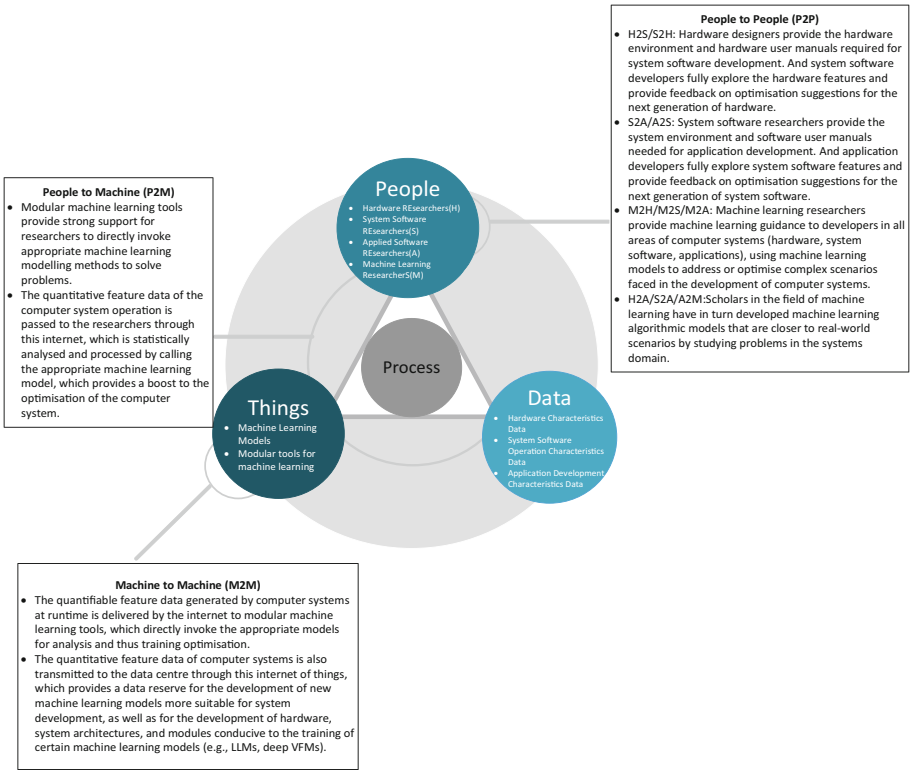


Fig. 5. P2P, P2M, and M2M in the ML-based computer system design and optimization

delivered by the internet to modular machine learning tools, which directly invoke the appropriate models for analysis and thus training optimization. A typical example is Synopsys technology, which launched the industry's first autonomous artificial intelligence application for chip design - design space optimization AI (DSO.ai). As an artificial intelligence and inference engine, DSO.ai can search for optimization goals in the vast solution space of chip design [31].

Computer systems' quantitative feature data is also transmitted to the data center through this connection(internet) of things, which provides a data reserve for the development of new machine learning models more suitable for system development, as well as for the development of hardware, system architectures, and modules conducive to the training of specific machine learning models (e.g., large language models, and visual foundation models).

The people-to-machine system also consists of two components, as shown in the leftmost section of Fig. 5: Modular machine learning tools strongly support computer system researchers' direct invocation of appropriate machine learning modeling methods to solve problems.

The quantitative feature data of the computer system operation is passed to the researchers through this connection(the internet). The data is statistically analyzed and

processed by calling the appropriate machine learning model, which boosts the computer system's optimization.

4 The Future of the IoE and Machine Learning and Computer System Development

In the future, IoE will become more mature, its applications will become more widespread, and machine learning and computer systems development will become unconventional when it comes to IoE.

4.1 The Future of Machine Learning in the IoE Era

Machine learning has developed tremendously in recent years. Its development trend can be divided into the following points: research on multimodal, larger scale, more parameters, and more versatile machine learning models; the combination of lightweight machine learning and edge computing; and the further application of machine learning in other scientific fields.

Multimodal, larger scale, more parameterized, and more general machine learning models will be a step forward. In the past few years, the language field has evolved from unimodal, multi-billion parameter models trained on tens of billions of token data (e.g., the 11-billion parameter T5 model) to multimodal, multi-billion or trillion parameter models trained on trillions of token data (e.g., OpenAI's 1.8-trillion parameter GPT-4 and Google's trillion-parameter Gemini multimodal large model) [40]. The growth in dataset and model sizes has resulted in significant improvements in accuracy on a wide range of tasks. It has been comprehensively demonstrated through performance optimizations on benchmark test sets widely used in developing large-scale language models [41].

Combining machine learning and edge computing has become particularly important with the rise of the Internet of Things and the widespread use of AI in mobile scenarios. On the one hand, in mobile scenarios, machine learning tasks require large amounts of data for training while at the same time requiring shorter response latency. In the case of autonomous driving, for example, a relatively large latency may significantly increase the risk of accidents. Thus, customized onboard computing devices must perform model inference at the edge [42]. Moreover, when many devices are connected to the same network, the adequate bandwidth is also reduced. Using edge computing can effectively reduce the competition between devices over communication channels. On the other hand, edge computing allows different edge devices to adopt customized learning tasks and models for the unused objects they are dealing with. For example, in the image recognition task in the security domain, the image information observed by video devices in different areas may vary greatly, so training only one deep learning model may not serve the purpose, and hosting multiple models on the cloud at the same time can be costly [43]. A more efficient solution is to train different models for each scene in the cloud and send the trained models to the corresponding edge devices.

In recent years, machine learning has grown in the basic sciences, from physics to biology. It has enabled many excellent real-world applications in related fields (e.g.,

materials science and medicine). For example, computer vision models are used to solve personal and global problems; they can assist doctors in their daily tasks, expand people's understanding of neurophysiology, and provide more accurate weather forecasts that can streamline disaster relief efforts [44]. Other machine learning models are becoming crucial in combating climate change by discovering ways to reduce emissions and increase alternative energy output.

4.2 The Future of Computer System Development in the IoE Era

The rapid development of the Internet of Things (IoT) brings new demands to developing computer systems. On the hardware side, miniaturization and low power consumption of hardware are further developed. IoT devices need more portable and low-power hardware design to adapt to various environments and application scenarios. Developing microprocessors and embedded systems makes IoT devices more portable and energy efficient. Meanwhile, new computing and storage architectures are emerging to handle the large amount of big data generated in the Internet of Everything, e.g., GPUs, heterogeneous processors, and AI processors [49, 50].

Along with the further maturity of the Internet of Everything, the era of ubiquitous computing, in which the human society, information space, and physical world are deeply integrated, is beginning. New scenarios integrating massive, heterogeneous, and heterogeneous resources of "people, machines, and things" are emerging, with an exponential increase in the complexity of the resources that need to be managed. It has become a development trend to build a ubiquitous operating system that manages all kinds of ubiquitous facilities/resources downstream and supports digital and intelligent applications upstream in all scenarios. Mei et al. [45] indicated that a new type of operating system, ubiquitous operating system (UOS), is emerging and in the exploratory period for the new mode and new scenarios of human-machine-object convergence ubiquitous computing in the future; Carlos F. Daganzo [46] also suggested that an ultra-long urban transport operating system that will make urban transport more convenient. New application models are giving rise to diverse application scenarios in operating systems. Mercedes-Benz [47] announced that 2024 models will have a new Mercedes-Benz operating system (MB.OS) MB.OS will be co-developed with Unity Technologies, focusing more on the 3D and connected experience, which will successively replace the MBUX infotainment platform that Mercedes-Benz models are now using. The different types of ubiquitous operating systems in the cloud-edge-end are more interactive and collaborative. Google [48] announced the release of a new open-source operating system, Kata OS, to provide a verifiable security system for embedded devices and improve user privacy and data security protection for IoT and embedded device operating systems.

5 Discussion and Conclusion

With the rapid development of the Internet of Everything (IoE), which provides favorable conditions for the development of machine learning, machine learning has made significant progress and solved many practical problems in computer system development. However, objectively speaking, the field of machine learning itself still has significant

challenges. Mainstream machine learning techniques are black-box techniques, making it impossible to predict hidden crises. To solve this problem, we need to make machine learning interpretable and intervenable. Most machine learning techniques, especially those based on statistics, rely heavily on probabilistic predictions and analyses based on data correlation acquisition. In contrast, rational human decision-making relies more on clear and plausible causal relationships derived from authentic and transparent factual causes and logically correct rule-based reasoning. Transitioning from using data correlation to solve problems to using causal logic between data to explain and solve problems is one of the core tasks that interpretable machine learning needs to accomplish. It is foreseeable that interpretable machine learning will drive further computer system development.

From the history of computer system development, the development of computer systems is closely linked to the IoE. The development of computer systems cannot be separated from IoE, and at the same time, the evolving computer systems also promote the development of IoE. The development of ML-based computer systems reflects people-to-people, people-to-machine, and machine-to-machine systems. Some of the future directions of existing computer systems are exploring new areas of IoE development, and we expect that the combination of the Internet of Everything and computer system development will bring discoveries.

References

1. Kiesler, N., Impagliazzo, J.: Perspectives on the internet of everything. In: Pereira, T., Impagliazzo, J., Santos, H. (eds.) IoECon 2022. LNCS, SITE, vol. 458, pp. 22–31. Springer, Cham (2023). https://doi-org-s.libyc.nudt.edu.cn:443/10.1007/978-3-031-25222-8_1
2. Leiner, B., et al.: A brief history of the internet. Comput. Commun. Rev. **39**, 22–31 (2009). https://doi.org/10.1145/1629607.1629613
3. Yang, L.T., Di Martino, B., Zhang, Q.: Internet of everything. Mobile Inf. Syst. **2017**, 1–3 (2017). https://doi.org/10.1155/2017/8035421
4. Angra, S., Ahuja, S.: Machine learning and its applications: a review. In: International Conference on Big Data Analytics and Computational Intelligence (ICBDAC), Chirala, Andhra Pradesh, India, pp. 57–60. IEEE (2017). https://doi.org/10.1109/ICBDACI.2017.8070809
5. Wang, H., Ma, C., Zhou, L.: A brief review of machine learning and its application. In: International Conference on Information Engineering and Computer Science (ICIEC), Wuhan, China, pp. 1–4. IEEE (2009). https://doi.org/10.1109/ICIECS.2009.5362936
6. Rumelhart, D., Hinton, G., Williams, J.: Learning representations by back-propagating errors. Nature **323**(99), 533–536 (1986). https://doi.org/10.1038/323533a0
7. Freund, Y.: Boosting a weak learning algorithm by majority. Inf. Comput. (1995). https://doi.org/10.1006/inco.1995.1136
8. Cortes, C., Vapnik, V.: Support vector networks. Mach. Learn. (1995). https://doi.org/10.1007/BF00994018
9. Hinton, G.E., Salakhutdinov, R.R.: Reducing the dimensionality of data with neural networks. Science **313**, 504–507 (2006). https://doi.org/10.1126/science.1127647
10. Rosenblatt, F.: The perceptron: a probabilistic model for information storage and organization in the brain. Psychol. Rev. **65**(6), 386–408 (1958)
11. Thomas, M.C., Peter, E.H.: Nearest neighbor pattern classification. IEEE Trans. Inf. Theory (1967). https://doi.org/10.1109/TIT.1967.1053964

12. Breiman, L.: Random forests. Mach. Learn. **45**, 5–32 (2001). https://doi.org/10.1023/A:1010933404324
13. MacQueen, J.B.: Some Methods for Classification and Analysis of Multivariate Observations (1967). https://api.semanticscholar.org/CorpusID:6278891
14. Ward, J.H.: Hierarchical grouping to optimize an objective function. J. Am. Stat. Assoc. **58**(301), 236–244 (1963). https://doi.org/10.1080/01621459.1963.10500845
15. Sibson, R.: SLINK: an optimally efficient algorithm for the single-link cluster method. Comput. J. Br. Comput. Soc. **16**(1), 30–34 (1973). https://doi.org/10.1093/comjnl/16.1.30
16. Defays, D.: An efficient algorithm for a complete-link method. Comput. J. Br. Comput. Soc. **20**(4), 364–366 (1977). https://doi.org/10.1093/comjnl/20.4.364
17. Dempster, A.P., Laird, N.M., Rubin, D.B.: Maximum likelihood from incomplete data via the EM algorithm. J. Roy. Stat. Soc. B **39**(1), 1–38 (1977). https://doi.org/10.1111/j.2517-6161.1977.tb01600.x
18. Ankerst, M., et al.: OPTICS: ordering points to identify the clustering structure. In: ACM SIGMOD INTERNATIONAL CONFERENCE on Management of Data, pp. 49–60. ACM Press (1999). https://doi.org/10.1145/304181.30418
19. Cheng, Y., Shift, M.: Mode seeking, and clustering. IEEE Trans. Pattern Anal. Mach. Intell. **17**(8), 790–799 (1995). https://doi.org/10.1109/34.400568
20. Shi, J., Malik, J.: Normalized cuts and image segmentation. IEEE Trans. PAMI **22**(8), 790–799 (2000). https://doi.org/10.1109/34.868688
21. Stratonovich, R.L.: Conditional Markov processes. Theory Prob. Appl. **5**(2), 156–178 (1960). https://doi.org/10.1137/1105015
22. Moussouris, J.: Gibbs and Markov random systems with constraints. J. Stat. Phys. **10**(1), 11–33 (1974). https://doi.org/10.1007/BF01011714
23. Lafferty, J., McCallum, A., Pereira, F.: Conditional random fields: probabilistic models for segmenting and labeling sequence data. In: Proceedings of the 18th International Conference on Machine Learning, pp. 282–289 (2001). https://dl.acm.org/doi/10.5555/645530.655813
24. Sutton, R.: Learning to predict by the methods of temporal differences. Mach. Learn. **3**, 9–44 (1988). https://doi.org/10.1007/BF00115009
25. Mnih, V., Kavukcuoglu, J., Silver, D., Graves, A.: Ioannis Antonoglou. Playing Atari with Deep Reinforcement Learning (2013). https://doi.org/10.48550/arXiv.1312.5602
26. Mnih, V., et al.: Asynchronous methods for deep reinforcement learning. In: Proceedings of the 33rd International Conference on International Conference on Machine Learning, pp. 1928–1937 (2016). https://dl.acm.org/doi/10.5555/3045390.3045594
27. Williams, R.J.: Simple statistical gradient-following algorithms for connectionist reinforcement learning. Mach. Learn. **8**(3), 229–256 (1992)
28. Silver, D., Lever, G., Heess, N., Degris, T., Wierstra, D., Riedmiller, M.: Deterministic policy gradient algorithms. In: Proceedings of the 31st International Conference on International Conference on Machine Learning, pp. 387–3395 (2014). https://dl.acm.org/doi/10.5555/3044805.3044850
29. Jonas, M.: On Bayesian methods for seeking the extremum and their application. IFIP Congress (1977)
30. Ren, H.: ParaGraph: layout parasitics and device parameter prediction using graph neural networks. In: Design Automation Conference (DAC), pp. 1–6 (2020). https://doi.org/10.1109/DAC18072.2020.9218515
31. Can AI improve chip design efficiency and reduce power consumption? (2022). https://zhuanlan.zhihu.com/p/537456547
32. Artificial intelligence drives the development of semiconductors (2023). https://zhuanlan.zhihu.com/p/667394848

33. Smith, W., Foster, I.T., Taylor, V.E.: Predicting application run times using historical information. In: Proceedings of the Workshop on Job Scheduling Strategies for Parallel Processing, pp. 122–142 (1998). https://dl.acm.org/doi/10.5555/646379.689526
34. Negi, A., Kumar, P.: Applying machine learning techniques to improve linux process scheduling. In: IEEE Region 10 Conference, Melbourne, Australia. IEEE (2005)
35. Ma, L., Van Aken, D., Hefny, A., Mezerhane, G., Pavlo, A., Gordon, G.J.: Query-based workload forecasting for self-driving database management systems. In: Proceedings of the 2018 International Conference on Management of Data. Presented at the SIGMOD/PODS 2018: International Conference on Management of Data (ICMD), pp. 631–645 (2018). https://dl.acm.org/doi/10.1145/3183713.3196908
36. Zhang, Y., Huang, Y.: "Learned": operating systems. ACM SIGOPS Oper. Syst. Rev. **53**(1), 40–45 (2019). https://doi.org/10.1145/3352020.3352027
37. Idreos, S., et al.: Learning Key-Value Store Design (2019). https://doi.org/10.48550/arXiv.1907.05443Focus to learn more
38. Zhang, J., Liu, Y., Zhou, K.: An end-to-end automatic cloud database tuning system using deep reinforcement learning. In: Proceedings of the 2019 International Conference on Management of Data (ICMD), pp. 415–432. ACM Comput (2019). https://doi.org/10.1145/3299869.3300085
39. Google just gave control over data center cooling to an AI (2018). https://www.technologyreview.com/2018/08/17/140987/google-just-gave-control-over-data-center-cooling-to-an-ai/
40. Jeff, D., Hassabis: All achievements of Google Research and Google DeepMind in 2023 (2023). https://finance.sina.com.cn/tech/2023-12-25/doc-imzzesth1127834.shtml
41. Google officially releases AI model Gemini (2023). https://www.oschina.net/news/269844/google-gemini-ai
42. Sasaki, K., Suzuki, N., Makino, S., Nakao, A.: Vehicle control system coordinated between cloud and mobile edge computing. In: 55th Annual Conference of the Society of Instrument and Control Engineers of Japan (SICE), Tsukuba, Japan, pp. 1122–1127 (2016). https://doi.org/10.1109/SICE.2016.7749210
43. Li, X., Li, J., Yiu, S., et al.: Privacy-preserving edge-assisted image retrieval and classification in IoT. Front. Comput. Sci. **13**, 1136–1147 (2019). https://doi.org/10.1007/s11704-018-8067-z
44. Kang, L., Chou, K., Fu, R.: Deep learning-based weather image recognition. In: 2018 International Symposium on Computer, Consumer and Control (IS3C), Taichung, Taiwan, pp. 384–387 (2018). https://doi.org/10.1109/IS3C.2018.00103
45. Mei, H., Cao, D., Xie, T.: Ubiquitous operating system: toward the blue ocean of human-cyber-physical ternary ubiquitous computing. Bull. Chinese Acad. Sci. **37**(1), 30–37 (2022). https://doi.org/10.16418/j.issn.1000-3045.20211117009
46. Daganzo, C.: An Operating System for Extra Long Urban Trains (2022). https://doi.org/10.1016/j.trb.2022.01.004
47. Mercedes Benz will introduce a new infotainment operating system starting from the 2024 model (2022). https://zhuanlan.zhihu.com/p/553317647
48. Google's open-source secure machine learning operating system, KataOS (2022). https://zhuanlan.zhihu.com/p/593068469
49. Chen, J.: More bang for your buck: boosting performance with capped power consumption. Tsinghua Sci. Technol. **26**(3), 370–383 (2020). https://doi.org/10.26599/TST.2020.9010012
50. Wang, R., et al.: Brief introduction of Tianhe exascale prototype system. Tsinghua Sci. Technol. **26**(3), 361–369 (2021). https://doi.org/10.26599/TST.2020.9010009