Generative AI

Subjects: Computer Science, Artificial Intelligence

Contributor: Carlos Eduardo Andino Coello , Mohammed Nazeh Alimam , Rand Kouatly

Generative AI models harness the capabilities of neural networks to discern patterns and structures within existing datasets and create original content. These AI models draw inspiration from human neuronal processes, learning from data inputs to create new output that matches learned patterns.

artificial intelligence ChatGPT GPT-3.5 GPT-4 Python programming

1. Introduction

In recent years, artificial intelligence (AI) has experienced exponential growth, marked by significant advancements in natural language processing and machine learning. This surge has brought about a transformation in various industries and applications. One specific area that has garnered considerable attention is AI-assisted programming. Advanced language models have the potential to revolutionize the way developers create, maintain, optimize, and test code.

OpenAI's release of the GPT models and the widely available ChatGPT represents a substantial breakthrough in the advancement of AI capabilities [1][2]. With each iteration, the models have demonstrated improved performance and versatility, generating increased interest in their potential uses and applications across multiple fields. In programming alone, these models have shown significant promise, particularly in automating tasks, improving code, and providing insights to developers.

The breakthrough in automated code generation has been significantly propelled [3] and greatly boosted by recent advancements in large language models like GPT-3 [4], surpassing the capabilities of earlier state-of-the-art deep learning methods [5].

As an illustration, OpenAI Codex [6], a refined iteration of GPT-3, can produce entirely accurate code for 29% of unfamiliar programming tasks using just one sample of generated programs. It was found that when testing 100 samples, 72% of them are correct. In [7], the authors evaluate the GPT Python code-writing capabilities and the correctness of the code generated. The results in this paper are based on only a small number of samples, which shows that the model can solve only 28% of the problems. Hammond et al. [8] investigated the possibility of using OpenAI Codex and other large language models (LLMs) to fix software security bugs. The results show that 67% of vulnerabilities in a selection of historical bugs in real-world open source projects can be fixed and discovered by LLMs. Meanwhile, Refs. [9][10] tested the usability of the code generated by LLMs and not the accuracy of the codes.

Xu and colleagues [11] compared the performance of code generated by GPT-Neo, GPT-J, and GPT-NeoX—all large language models (LLMs)—when trained with a substantial number of parameters derived from ready codes in 12 different languages. Zan et al. [12] investigated the existing large language models for NL2Code and summarized them from diverse perspectives. However, neither of these research studies investigated the accuracy and the quality of the code generated by LLMs.

2. Generative AI History

Modern generative AI development began in the 1940s after the conception of the first artificial neural networks (ANNs). However, due to constraints such as limited computational capabilities and insufficient knowledge of the brain's biological workings, ANNs failed to draw significant interest until the 1980s. During this period, parallel advances in hardware and neuroscience, along with the emergence of the backpropagation algorithm, eased the training process of ANNs. Previously, training NNs was a demanding task, as there was no effective method to compute the error's gradient relating to each neuron's parameters or weights. However, backpropagation automated the training procedure, unlocking the potential usage of ANNs [13].

In 2013, Kingma and Welling presented a novel model structure named variational autoencoders (VAEs) in their papers entitled "Auto-Encoding Variational Bayes". VAEs are generative models grounded in the principle of variational inference. They offer a mechanism for learning via a condensed representation of data, where the data are transformed into a lower-dimensional area called the latent space through an encoding process. Then, the decoder component reconstructs the data back into their original data space [14].

In 2017, Google researchers introduced a pivotal development in their research titled "Attention Is All You Need". This new architecture, called Transformer, was a revolution in language generation [15]. Unlike previous language models based on long short-term memory (LSTM) [16] or recurrent neural networks (RNN) frameworks [17], Transformer allowed for parallel processing while retaining context memory, leading to superior performance [17].

In 2021, OpenAI released a fine-tuned version of GPT, Codex, which was trained on code publicly available on GitHub. Early results showed that the fine-tuned model was able to solve around 30% of the Python problems used, compared to the 0% that the current GPT version (GPT-3) was able to achieve. This served as an early look into how large language models (LLMs) [10] can learn and generate code. Codex then served as the basis for GitHub Copilot [10].

GitHub Copilot is an AI programming tool that can be installed in the most popular code editors and is powered by GPT-4. It reads the code and can generate suggestions and even write code instantly. In a controlled test environment, researchers found that programmers who used Copilot finished tasks approximately 55.8% quicker than those who did not, speaking to the potential of AI tools in programming [18][19].

In another research work, GPT's Python code generation is deemed remarkable, showing that it can help novice programmers to solve complex coding problems using only a few prompts. However, both studies have shown that human input is almost always required to steer ChatGPT in the correct direction [20].

3. What Is Generative AI

Generative AI models harness the capabilities of neural networks to discern patterns and structures within existing datasets and create original content [21]. These AI models draw inspiration from human neuronal processes, learning from data inputs to create new output that matches learned patterns. This involves advanced techniques that range from generative adversarial networks (GANs) [21], large language models (LLMs), variational autoencoders (VAEs), and transformers to create content across a dynamic range of domains [22].

Numerous methodologies, such as unsupervised or semi-supervised learning, have empowered organizations to utilize abundant unlabeled data for training and laying foundations for more complex AI systems. Referred to as foundation models, these systems, which comprise models like GPT-3 and Stable Diffusion, serve as a base that can be proficient in multiple tasks. They enable users to maximize the potency of language, such as constructing essays from brief text prompts using applications like ChatGPT or creating remarkably realistic images from text inputs with Stable Diffusion [23].

Generative AI models can refine their outputs through repeated training processes by studying the relationships within the data. They can adapt parameters and diminish the gap between the intended and created outputs, continually enhancing their capacity to produce high-quality and contextually appropriate content. The utilization of this technology is often initiated with a prompt, followed by iterative exploration and refining of variations to guide content generation [24].

References

1. Brown, T.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J.D.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A.; et al. Language models are few-shot learners. Adv. Neural Inf. Process. Syst. 2020, 33, 1877–1901.
2. Fan, L.; Li, L.; Ma, Z.; Lee, S.; Yu, H.; Hemphill, L. A bibliometric review of large language models research from 2017 to 2023. arXiv 2023, arXiv:2304.02020.
3. Ni, A.; Iyer, S.; Radev, D.; Stoyanov, V.; Yih, W.T.; Wang, S.; Lin, X.V. Lever: Learning to verify language-to-code generation with execution. In Proceedings of the International Conference on Machine Learning 2023, Honolulu, HI, USA, 23–29 July 2023; PMLR: Westminster, UK; pp. 26106–26128.
4. OpenAI; Pilipiszyn, A. GPT-3 Powers the Next Generation of Apps. 2021. Available online: https://openai.com/blog/gpt-3-apps/ (accessed on 26 November 2023).

5. Hardesty, L. Explained: Neural Networks, MIT News. Massachusetts Institute of Technology. 2017. Available online: https://news.mit.edu/2017/explained-neural-networks-deep-learning-0414 (accessed on 25 July 2023).
6. Zaremba, W.; Brockman, G.; OpenAI. OpenAI Codex. 2021. Available online: https://openai.com/blog/openai-codex/ (accessed on 23 November 2023).
7. Austin, J.; Odena, A.; Nye, M.; Bosma, M.; Michalewski, H.; Dohan, D.; Jiang, E.; Cai, C.; Terry, M.; Le, Q.; et al. Program Synthesis with Large Language Models. arXiv 2021, arXiv:2108.07732.
8. Pearce, H.; Tan, B.; Ahmad, B.; Karri, R.; Dolan-Gavitt, B. Can OpenAI Codex and Other Large Language Models Help Us Fix Security Bugs? arXiv 2021, arXiv:2112.02125.
9. Vaithilingam, P.; Zhang, T.; Glassman, E.L. Expectation vs. experience: Evaluating the usability of code generation tools powered by large language models. In Proceedings of the Chi Conference on Human Factors in Computing Systems Extended Abstracts, New Orleans, LA, USA, 29 April–5 May 2022; pp. 1–7.
10. Chen, M.; Tworek, J.; Jun, H.; Yuan, Q.; Pinto, H.P.D.O.; Kaplan, J.; Edwards, H.; Burda, Y.; Joseph, N.; Brockman, G.; et al. Evaluating large language models trained on code. arXiv 2021, arXiv:2107.03374.
11. Xu, F.F.; Alon, U.; Neubig, G.; Hellendoorn, V.J. A systematic evaluation of large language models of code. In Proceedings of the 6th ACM SIGPLAN International Symposium on Machine Programming, San Diego, CA, USA, 13 June 2022; pp. 1–10.
12. Zan, D.; Chen, B.; Zhang, F.; Lu, D.; Wu, B.; Guan, B.; Yongji, W.; Lou, J.G. Large language models meet NL2Code: A survey. In Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics 2023, Toronto, ON, Canada, 9–14 July 2023; Volume 1, pp. 7443–7464.
13. Basheer, I.A.; Hajmeer, M. Artificial neural networks: Fundamentals, computing, design, and application. J. Microbiol. Methods 2000, 43, 3–31.
14. Kingma, D.P.; Welling, M. Auto-encoding variational bayes. arXiv 2013, arXiv:1312.6114.
15. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, Ł.; Polosukhin, I. Attention is All You Need. In Advances in Neural Information Processing Systems; MIT Press: Cambridge, MA, USA, 2017; p. 30.
16. Azzouni, A.; Pujolle, G. A long short-term memory recurrent neural network framework for network traffic matrix prediction. arXiv 2017, arXiv:1705.05690.
17. Sherstinsky, A. Fundamentals of recurrent neural network (RNN) and long short-term memory (LSTM) network. Phys. D Nonlinear Phenom. 2020, 404, 132306.

18. Peng, S.; Kalliamvakou, E.; Cihon, P.; Demirer, M. The Impact of AI on Developer Productivity: Evidence from GitHub Copilot. arXiv 2023, arXiv:2302.06590. Available online: http://arxiv.org/abs/2302.06590 (accessed on 20 August 2023).
19. GitHub. Copilot Your AI Pair Programmer. Available online: https://github.com/features/copilot (accessed on 13 August 2023).
20. Poldrack, R.A.; Lu, T.; Beguš, G. AI-assisted coding: Experiments with GPT-4. arXiv 2023, arXiv:2304.13187. Available online: http://arxiv.org/abs/2304.13187 (accessed on 23 July 2023).
21. NVIDIA. What Is Generative AI? 2023. Available online: https://www.nvidia.com/en-us/glossary/data-science/generative-ai/ (accessed on 23 July 2023).
22. Shanahan, M. Talking about large language models. arXiv 2022, arXiv:2212.03551.
23. Wang, K.; Gou, C.; Duan, Y.; Lin, Y.; Zheng, X.; Wang, F.Y. Generative adversarial networks: Introduction and outlook. IEEE/CAA J. Autom. Sin. 2017, 4, 588–598.
24. Elastic. What Is Generative AI?|A Comprehensive Generative AI Guide. 2023. Available online: https://www.elastic.co/what-is/generative-ai (accessed on 23 July 2023).

Retrieved from https://encyclopedia.pub/entry/history/show/124757