iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 11 **Issue:** II **Month of publication:** February 2023

DOI: https://doi.org/10.22214/ijraset.2023.49048

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 11 Issue II Feb 2023- Available at www.ijraset.com

Open AI Codex: An Inevitable Future?

Aman Kumar[1], Priyanka Sharma[2]
[1]*Student,* [2]*Assistant Professor, Manav Rachna International Institute Of Research And Studies, Faridabad, Haryana,India*

Abstract: OpenAI Codex is an AI system that converts natural language into code, OpenAI shows how the software can be used to build simple websites and rudimentary natural language games, translate between different programming languages, and answer data science queries. The user enters English commands into the software. For example, Create a web page with a menu on the side and a title at the top and Codex will translate this into code.
Codex is the model behind GitHub Copilot, developed and launched in partnership with GitHub. Codex is proficient in over 10 programming languages and can interpret and execute simple natural language commands on your behalf, allowing you to create natural language interfaces to your existing applications.
OpenAI Codex is based on GPT-3, GPT-3 is an auto regressive language which has capability to convert text to image, face recognition, handwriting recognition, translation, sentence analysis, intelligent recommend-er. This core ability of gpt-3 is used to develop Open-AI Codex.

I. INTRODUCTION

If you've ever written code, you may have noticed that even complex software can (and should) be broken down into smaller units, such as functions, that perform common tasks.
Creating these small pieces is very cumbersome. It's not particularly difficult or interesting in many cases, but it takes time to create and distracts you from the tricky parts of the software development process.
OpenAI's Codex promises to create simple functions based on natural language descriptions of functionality. Developers only need to write function signatures and docstrings, and Codex handles the implementation.
What the Codex does isn't all that new. It's just a language model specialized for writing code. In fact, it is a direct descendant of GPT-3. If you've never heard of GPT-3, or are unfamiliar with what GPT-3 is, I recommend skimming through this introductory blog post before proceeding. Codex is basically a fine-tuned version of GPT-3.

II. OBJECTIVES OF THE STUDY

This research aims to see how codex works and performs with the currently available hardware technologies and assess the future of codex in computer systems design and related services.
We will also cover the capabilities of codex to see what its capable of and how it performed during its early test phase Due to the way the codex works we will also evaluate the legal implications of codex to determine whether the technology is ethical or not. With every new technological innovation comes unavoidable limitations and codex is no exception. We will also cover the various limitations that comes with tech in this paper. And in the end we will evaluate the future scope of the codex and areas of improvement to make this more viable for it to go mainstream.

III. OPENAI CODEX

Codex is based on GPT-3, OpenAI's language generation model. GPT-3 has been trained on the vast majority of the web to be able to generate and parse written words in an impressive way. One of the uses the users have found for GPT-3 was code generation, but Codex improves upon its predecessor and is specially trained on open-source code repositories scraped from the internet.
OpenAI Codex is a descendant of GPT-3, its training data contains both natural language and billions of lines of source code from publicly available sources, including code in public GitHub repositories.
OpenAI Codex is proficient in over a dozen languages including JavaScript, Python, Go, Perl, PHP, Ruby, Swift and TypeScript, and even Shell OpenAI used an earlier version of Codex to create a tool called Copilot for GitHub, a code repository owned by Microsoft, a close partner of OpenAI. Copilot is similar to Gmail's autocomplete tool, suggesting ways to complete a line of code that the user types. However, the new version of OpenAI's Codex is much more advanced and flexible in creating code than just completing it.

A. *Working*

Codex creates this code through training on all public code available on GitHub and other databases (totaling 159 GB) as well as our own knowledge of the coding context Which is then used in codebase as a "text" corpus to train Codex for the following word prediction language modeling tasks. You can also create generic code embedded in your code. How is the tool supposed to know what the helicopter is? It can infer what the object is based on other usages and contexts, even if the user doesn't define it.
Using the API requires thought and trial and error. It doesn't make non-programmers programming experts, but it's certainly a step in the right direction. It's not a direct replacement for programmers but more of an aid, it takes the hassle out and makes coding faster and more accessible.

B. *Capability*

OpenAI started with a pre-trained GPT-3 model. The team then collected Python code files from his 54 million public GitHub repositories and filtered them into a final dataset of 159 GB. This model uses the same text tokenizer as GPT-3. However, researchers found this to be suboptimal because the distribution of words in code differs from that in natural language. Additionally, Python code contains a large amount of whitespace, so the team introduced an additional set of tokens to represent whitespace "runs." The team manually created HumanEval, an open-source test dataset of 164 programming problems consisting of model prompts and a set of unit tests to verify the validity of the generated code. When Codex generated one solution per problem, 28.7% of the problems passed the unit tests. When Codex was allowed to generate 100 solutions for each problem, it generated at least one correct result for 77.5% of the problems.

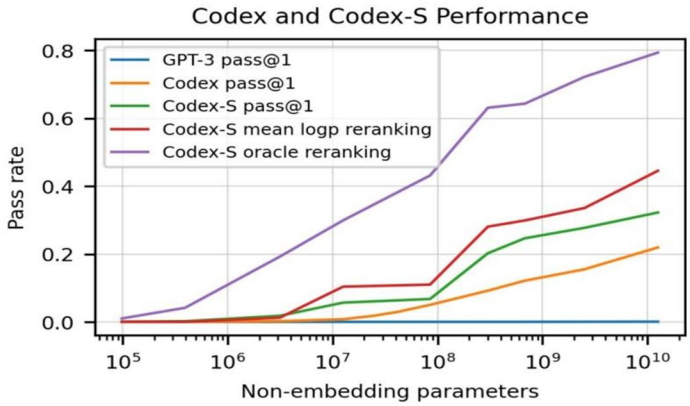


Fig.1 Performance comparison (17)

Other previous code generation models in the literature are often evaluated using fuzzy matching of outputs to reference outputs. BLEU score. In contrast, the OpenAI team chose to evaluate functional correctness, arguing that this is how human developers judge code. The specific metric used is pass@k. This means that if the model generates k code samples and the samples pass the unit tests, the model has solved the problem. The 12B-parameter Codex model was 28.8% for k=1 and 28.8% for k=100, compared to 2.58% and 7.59% for TabNine's largest free model, and 11, 6%, and 27.74% for GPT-J. achieved a score of 72.31%. Codex itself can speak more than a dozen coding languages, including JavaScript, Go, Perl, PHP, Ruby, Swift, and TypeScript. It's most proficient, though, in Python. Codex can also control other programs. In his demo it shows how the software can be used to create a voice interface for Microsoft Word. Since Word has its own API, Codex can enter instructions into code built from the user's voice commands. They tested the codex by copying the poem into a Word document and tells Word (via Codex) to first remove all indentation, then number the lines, and count the frequency of certain words.

IV. LIMITATIONS

While codex can generate the complex code via string commands specified by the user it still cannot generate the high-level code but to recreate the "Average" code found on GitHub. This means that the Codex has adopted and perpetuated common bad programming habits. (This problem is similar to language models preserving stereotypes present in training data.) Several qualitative metrics were developed that measure the capabilities of the code generation model while controlling the complexity and level of abstraction of the specification.

ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 11 Issue II Feb 2023- Available at www.ijraset.com

If you apply this framework, you will see that Codex can suggest syntactically incorrect or undefined code and call functions, variables and attributes that are undefined or out of scope of the required codebase. Additionally, Codex struggles to parse the through his ever-lengthening high-level or system-level specification.

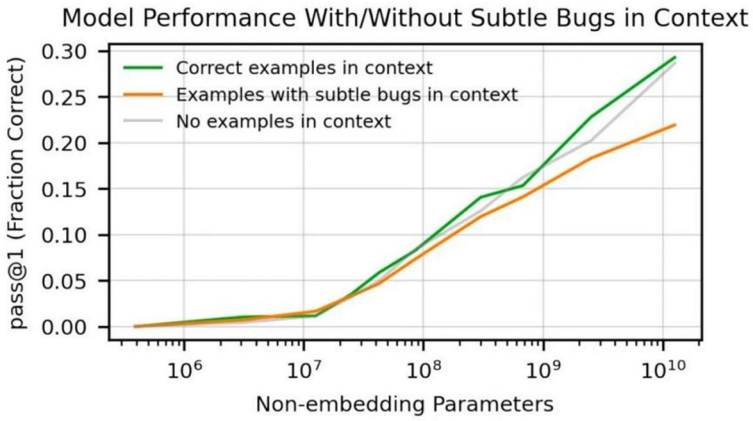


Fig.1 Performance comparison with bugs (17)

To demonstrate this 13 basic building blocks were created which modifies the input string to deterministic way like “converting the input string to lowercase ” or “removing every second character from the input string ”. it was found that the number of chained link docstring increased which decreased the model performance exponentially.

The Calculation Scale has also become an essential item for progression. With the test runs it was established that more computational power needs to be added to improve the performance for ever-larger datasets. In 2018 OpenAI Already pushed the limits of practical computing in gpt-1. the previous model could fit in one GPU because the model itself was small but its not the same case for codex, computer manufacturer and cloud operator Lambda Computing predicts that running that much processing power on a single GPU would take 355 years, or $4.6 million at the price of a standard cloud GPU. With the current technology of codex As long as building bigger and bigger models continues to be the route the more the processing power would be required and this whole system is only dependant on moore’s law to be viable

A. Legal Limitations

There are some legal considerations associated with generated codes. First, training an AI system on
internet data preliminary research also revealed that Codex models rarely produced
code that was identical to the content of the training data. Such incidence was less than 0.1% in a study examining the frequency of code generation matching code snippets of code in training data . In these rare cases, the generated code consisted of common expressions or rules within the programming language that appeared repeatedly in the training data. To the extent that the generated code looked the same as the training data but it was due to the prediction weights of the model rather than saving and copying specific code. OpenAI unfairly benefits from the published work of others. For example, GitHub Copilot's knowledge base consists primarily of code written by others, ultimately completed through a library of open source works originally created for personal benefit. . The same applies to Codex, but according to OpenAI, this data is legally protected through fair use.

V. CAUTION ADVISED

GPT-3, a general-purpose audio transformer that powers the Codex, was recently released to the public, but the Codex itself remains a technical preview open to limited user selection. Codex supports his Copilot feature on GitHub. It's a programming assistant available as a plugin for Visual Studio Code that can provide AI-powered autocompletion and code conversion on the fly.

Its skills are still raw, but it gives a good idea of what the future holds for programmers, computer scientists, and of course, malicious users. When such systems inevitably become a disruptive element in the day-to-day work of computer engineers, of course, what impact does this have on the activities of cybercriminals? With that in mind, we tested the range of Codex capabilities, focusing on the most typical aspects of cybercriminals: intelligence, social engineering, and exploitation. It's probably not the first time you've wondered what happens to all the information in public repositories after being filtered through a fine mesh of GPT-3 neural networks.

The initial issues with Copilot's proposal for copyrighted snippets of code had already surfaced, but whether sensitive information existed in his GPT-3 knowledge base and exploiting Codex's code generation I wanted to see if I could steal it by using
Personal and confidential information is disclosed through code.
Public repositories can become treasure troves of sensitive data waiting to be discovered by malicious actors. In our testing, we found that Codex can be tricked into disclosing sensitive data that remains in the repository by generating code that ultimately requires access to the data.

VI. FUTURE SCOPE

There are some promising directions to improve the consistency of the current code generation model, which could also greatly improve the usability of the model. A starting point is to curate records more carefully and then train them to remove buggy or unsafe code. Another option is to label the pre-training data based on code quality and vote for models labeled as high quality at deployment. A common way to tune Transformer behavior is to tune your dataset with high-quality, bug-free code. However, writing error-free code is notoriously difficult for most people. Formal analysis or other Code quality metrics are used to filter the input records instead of retrieving them by labeling.It has been successfully applied to the language model, improving consistency and performance. In the context of code models, the point is to collect data from code written by real humans about whether the generation was correct and useful. Fully customizing the model is a difficult human problem. Especially if the model is more knowledgeable or capable than the supervisor. It is also difficult to tell if the model is perfectly tuned or if the tuning metric needs more work. Consistency cannot be determined by input/output behavior alone, but we need transparency tools that can understand the model well enough to determine consistency.

VII. LIMITATIONS OF THIS PAPER

This paper is somewhat limited due to the availability of Open AI Codex to the general public, right now codex is invite only which means only few people used codex outside the team which made it difficult for the author of this paper to analyze and experience the tech himself. Based on current trends, a statistical analysis using Spearman's correlation coefficient to account for abnormal data, shows there is a direct positive correlation between the availability of the codex and this paper.
Henceforth, I am unable to articulate the further expansion of this paper due to the limitation of my restricted knowledge.Sincere apologies as my highly incoherent candid of a research paper has been quite the inadequate material due to my poor insufficient comprehension of the topic at hand.Further analysis is left as an exercise to the reader

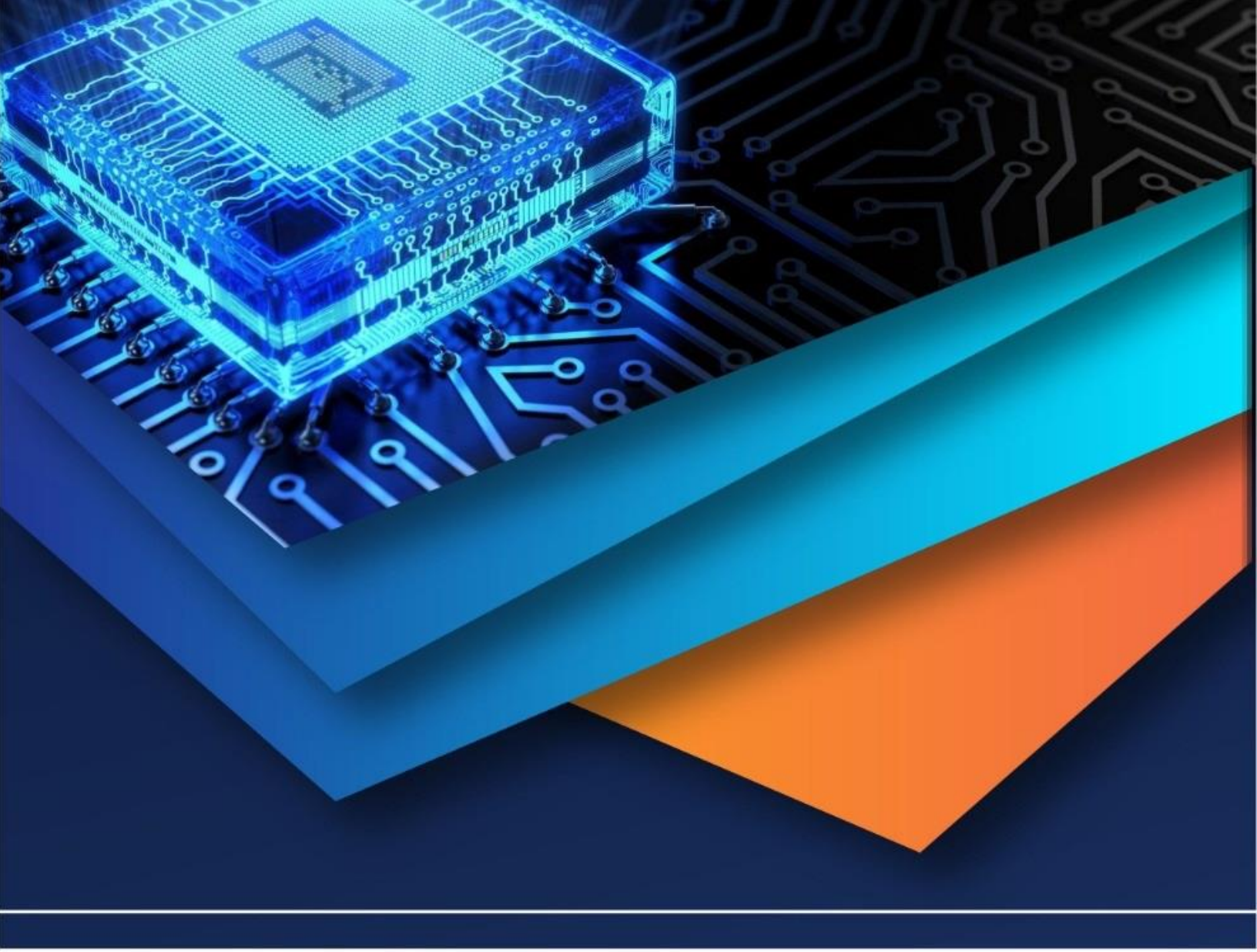
VIII. CONCLUSION

Codex allows non-coders to dive in, try things out, and bring their ideas to life. Artists, teachers, and many people from professions unrelated to language programming develop their own approaches and bring their visions to life.
It also helps programmers quickly try and break brainstorming experiments instead of spending hours on "just try" programming.
Chris Dillon from the codex team said "Any version of this is going to enable self improvement. We have just built a hammer for the world to play with. Maybe it can't exactly hammer on itself yet but it's going to be an accelerator. The first step is an AI assistant. This is great, exactly what I always imagined. I can't wait to combine this with other automation."
But codex also has serious problems as it scours the public repository for its code generation which may or may not always be correct or clean code resulting in bad result code, it also creates a dilemma on how ethical the tech really is since it is directly profiting from the work of others as it is taking the already build code from publicly available sources for its code generation. Codex team claims it is protected under fair use but it still raises the question of academic integrity as it is basically a form of plagiarism. This codex tech still has a long way to go for it to be considered a viable tool in our day-to day life, right now its more of a dumb coding assistant which you can use to get help if you are stuck or need some reference code. The analysis is severely limited by my lack of understanding of what I am doing

REFERENCES

[1] Forward-Looking Threat Research Team. Codex Exposed: Exploring the Capabilities and Risks of OpenAI's Code Generator,URL https://www.trendmicro.com/en_us/research/22/a/codex-exposed--exploring-the-capabilities-and-risks-of-openai-s-.html
[2] Alberto Romero.How OpenAI Codex Will Revolutionize Programming, URL https://towardsdatascience.com/heres-how-openai-codex-will-revolutionize-programming-and-the-world-e8432aafc5f7
[3] Abby Braden.OpenAI's Codex Turns Natural Language to Computer Code, URL https://technologyadvice.com/blog/information-technology/openai-codex/
[4] Tanushree Shenwai.OpenAI Releases New Version of GPT-3 and Codex That Can Edit or Insert Content Into Existing Text,URL https://www.marktechpost.com/2022/03/21/openai-releases-new-version-of-gpt-3-and-codex-that-can-edit-or-insert-content-into-existing-text%EF%BF%BC/

[5] Ben Dickson.OpenAI Codex shows the limits of large language models,URL https://venturebeat.com/business/openai-codex-shows-the-limits-of-large-language-models/
[6] Merzmensch.Codex by OpenAI, in Action, URL https://towardsdatascience.com/codex-by-openai-in-action-83529c0076cc
[7] James Finnie-Ansley,Paul Denny,Brett A. Becker,Andrew Luxton-Reilly and James Prather.The Robots Are Coming: Exploring the Implications of OpenAI Codex on Introductory Programming,URL https://dl.acm.org/doi/fullHtml/10.1145/3511861.3511863
[8] James Vincent.OpenAI can translate English into code with its new machine learning software Codex, URL https://www.theverge.com/2021/8/10/22618128/openai-codex-natural-language-into-code-api-beta-access
[9] Fabian Gringel.OpenAI Codex: Why the revolution is still missing, URL https://dida.do/blog/codex
[10] https://en.wikipedia.org/wiki/OpenAI_Codex
[11] https://gpt3demo.com/apps/openai-codex
[12] https://openai.com/blog/openai-codex/
[13] https://beta.openai.com/docs/introduction/key-concepts
[14] https://beta.openai.com/docs/models
[15] https://en.wikipedia.org/wiki/OpenAI
[16] Roy Maria John.Introduction to Open AI – Codex, URL https://blogs.infosys.com/digital-experience/emerging-technologies/introduction-to-open-ai-codex.html
[17] Evaluating Large Language Models Trained on Code, URL https://doi.org/10.48550/arXiv.2107.03374
[18] https://github.com/features/copilot
[19] https://python.plainenglish.io/generate-python-code-with-openai-codex-api-9617f8acd7bd
[20] https://learn.microsoft.com/en-us/azure/cognitive-services/openai/how-to/work-with-code
[21] Frank F. Xu, Uri Alon, Graham Neubig, Vincent J. Hellendoorn. A SYSTEMATIC EVALUATION OF LARGE LANGUAGE MODELS OF CODE, URL https://arxiv.org/pdf/2202.13169.pdf
[22] https://codeforces.com/blog/entry/94353?f0a28=1



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY