Reimagining the Roles of Software Developers During the Emergence of AI

Mr. Vinayak Arora
Research Scholar
Invictus International Amritsar

Dr. Saurabh Grover
Professor
Invictus International Amritsar

Abstract

Artificial Intelligence broadly is the ability of a digital computer to carry out rational thinking, problem solving and carry out tasks associated with human beings. At the current rate of progression AI is gaining more steam and momentum. Many people, especially those who are directly involved in programming or mainly deal with programming, are dealing with increased stress due to the fact that their jobs might be replaced, and for good reason. At this point of time AI is a very beneficial tool for many software engineers, however, AI technologies have only started advancing and will continue to do so at a very high rate. We already have technologies, with the use of generative AI assist developers by writing the code for them. These AI toolings might also not be for the primary benefit of new programmers. It may encourage them not to really learn the language or the basics of what they are trying to learn, but instead grow reliant on these technologies. Brett A. Becker, Paul Denny and James Finnie-Ansley also suggested that these new technologies might shfit the landscape from code learning to code reading and evaluation, making the higher level design decisions instead of humans writing the code themselves. Given some time to mature, AI will be able to do a majority of the tasks required in a profession such as a programmer and with more efficiency. Ford, Martin in a study suggested that even though AI might not cause immediate job disruption, it might still be able to create a future unemployment crisis. We also have to be apprehensive of the fact that these companies often use publically available code to train their models to write code. Even though the code is publically available, it does not give them the consent to use this publically available code to train their models without the express permission of the author.

Key words – Artificial Intelligence, impact, coding, data

Introduction

Artificial Intelligence broadly is the ability of a digital computer to carry out rational thinking, problem solving and carry out tasks associated with human beings.

At the current rate of progression AI is gaining more steam and momentum. Many people, especially those who are directly involved in programming or mainly deal with programming, are dealing with increased stress due to the fact that their jobs might be replaced, and for good reason. At this point of time AI is a very beneficial tool for many software engineers, however AI technologies have only started advancing and will continue to do so at a very high rate. We already have technologies, with the use of generative AI assist developers by writing the code for them.

These AI toolings might also not be for the primary benefits of new programmers. It may encourage them not to really learn the langauge or the basics of what they are trying to learn, but instead grow reliant on these technologies. Brett A. Becker, Paul Denny and James Finnie-Ansley also suggested that these new technologies might shfit the landscape from code learning to code reading and evaulation, making the higher level design decision instead of humans writing the code themselves.

Review of literature

In 2020 Tandon, Divya, Jyotika Rajawat, and Monisha Banerjee found that AI has a lot of benefits and shortcomings in the field of dentistry. It can result in increased productivity due to its efficiency, makes accurate diagnosis and the results can also be standardized. The data used for training the models can result in “data snooping bias”, is very expensive and has a lot of system complexity.

In 2020 Alexander, Alan, et al in their study found that investments in the field of AI based medical imaging have grown exponentially since their last review in 2018. The investments have doubled to almost $1.17 billion dollars and the companies have tripled to around 113. Despite this level of adoption, the current AI solutions are not heavily adopted and are only used for limited conditions. Adoption by physicians and approval by regulatory bodies is currently holding back the adoption of these technologies. Ai is likely to also gain the CT market due its enhancement of imaging quality.

In 2019 Wasilow, Sherry, and Joelle B. Thorpe asserted that the incorporation of AI and robotics into the military raises its potential ethical concerns to a much higher level. The inclusion of AI into the military will need to be a thoughtful and inclusionary process. More ethical questions should be posed in order. The supporters and opponents of these technologies or group of technologies should also come to a agreement on what these technologies imply and include. However the reasons to invest in this sector are also vast.

In 2019, Majd Latah and Levent toker found an increased adoption of AI techniques to solve a vast variety of networking problems. They also adressed new challenges introudced by Software-defined Networking paradigms. Using AI has proved to be a very useful tool in Software-defined networking.

In 2019 Holzinger, Andreas, et al found that the AI models of today are "black box" models and lack an explicit declarative representation of knowledge. We might also loose our control in the human-AI relationship.

In 2017 Etzioni, Amitai, and Oren Etzioni implied that the many values held by us as a species are incorporated into us by law or by informal social control. AI use cases such as driverless cars should abide by the law and abide by these legal values or be taken off the road. In case of damage the AI's programmer, and manufacturers would need to be held liable.

In 2017 Hamet, Pavel, and Johanne Tremblay suggested the possibility of AI tooling becoming more sophisticated and surpassing humanity, but by making AI tools open source and with proven clinical utility, society will be able to accumulate its benefits.

In 2017 Bundy, Alan found that AI can have several advantages and disadvantages. It can increase productivity and make services and products more widely available. Humans aided with AI can make higher qualiimpressive expert systems that performed
well in narrow domains but were brittle and broke easily. Machine learning burgeoned in the 1990s as more data started becoming available and powerful predictive systems emerged. But we now know that prediction isn'tenough. Predictive models don't tell us why people are dying. Indeed, a major technical challenge for AI is to unify symbolic AI, which devoted a considerable amount of effort to explanation, with current day machine learning, which is relatively weak on that frontty education more widely

available. Some features such as security, privacy and safety need to be implemented into the future AI models. Humans need to be more careful and accountable for there wrong doings and test things more thoroughly rather than trusting AI technology blindly.

In 2016 Dhar, Vasant found that the law and regulation are filled with concepts which are deeply human such as intent and emotion and we are not equipped with handling robots and algorithms which do not have any of these qualities. Robots and algorithms are also new territory with no previously set precedent. Other barriers also exist such as the fact that the current AI models are largely predictive in nature and exist only because they are trained on a large set of data. The other form of training the models such as unsupervised learning is also a challenge.

In 2016, rajesh H. Kulkarni and Polacholla Padmanbham proposed an architecture to organise and extend software engineering process models. They proved that IOI metrics have a better correlation than UGAM for better waterfall models.

In 2014, Bhagyashree W.Sorte, Pooja P. Joshi, and Prof. Vandana Jagtap concluded that there is a scope for automation of software development life cycle. Artificial Intelligence, Expert Systems, and Knowledge Engineering will also continue to play a major role in automating many activities associated with software development.

In 2004 Ramesh, A. N found that despite the early optimism regarding these technologies, clinicians are not keen in letting in technology in the decision making process. There must be work done by researchers to produce evidence for use of technology in the field, and these technologies will provide a boost to the clinicians of tomorrow.

Overview of Ai subfields

1. Machine Learning → Machine Learning is currently one of the most used ways to develop Artificial Intelligence. In it the machine is left to develop its own algorithms without being explicitly programmed with a pre-existing algorithm in mind. We currently have a few techniques to properly develop Machine Learning models.

-Supervised Machine Learning uses labeled data sets to train the model. The task while creating a Supervised Machine Learning model is to create a estimator which will predict the label of the input values provided to it. The model is provided with a particular data set of objects with a set of labels attached to each object, the model learns by comparing the object with the label and associating properties of the object with the label. Supervised Machine Learning is most commonly used in classification problems such as classifying emails as spam or not spam. Some algorithms used in Supervised Machine Learning are as follows:

-- Decision Trees → Decision Trees are Machine Learning algorithms which can be used to classify data. Let's say we have three conditions A, B and C along with an object X, if Condition A is true X can be classified as J, if B is true and A is false we can classify the X as K but if both A and B are false and C is only true we can classify X as L. Decision Trees at heart can be represented with a few if else statements.

- Unsupervised Machine Learning works by providing the algorithm with un-labeled data sets. This feature allows the algorithm to discover hidden patterns which would not be otherwise missed by humans, it then classifies the data based on that pattern. There are no labels and no right or wrong answers, there is just the data. Some algorithms for unsupervised machine learning are

-- K means clustering → This is one of the simplest Unsupervised ML algorithm. This algorithm works by identifying clustered data points present within a dataset, it then identifies a "K centre" present within the cluster and groups the cluster together and finds some common attributes within this K cluster.

-- Principal Component Analysis → Principal Component Analysis is most commonly used for larger data sets with high number of dimensions. The algorithm reduces the dimensions and transforms it into a set of linearly uncorrleated variables known as the principal components.

- Reinforcment Machine Learning algorithms do not require any labled or unlabeled data sets at all, it learns by experience. Reinforcement Machine Learning algorithms try to maximise the potentital reward for any situation. For any decision it takes, the algorithm is rewarded with positive feedback for desirable behavior and negative feedback for un-desirable behaviour.

- Neural Networks are a collection of connected nodes which somewhat model the neurons of the human brain. The nodes or neurons are layered in multiple ways, the input is taken and processed by multiple hidden layers and then the output layer displays or sends out the output.

- Deep Learning utilizes Neural Networks to build algorithms. There must be three or more hidden layers present. Unlike most other Machine Learning algorithms which require structured data sets, Deep Learning algorithms can ingest and utilize unstructured data.

2. Natural Language Processing → Natural Langugage Processing is a field within Artificial Intelligence which is focused on understanding and processing human speech and language. It combines multiple Machine Learning, Deep Learning, and statistical models to achieve this goal. Common applications of this model are in chat bots, digital assisstants etc.

- Large Language Models → These are deep learning algorithms which can recognize, summarize, predict and generate text. These models usually contain a lot of parameters which are necessary for training. They are trained on a vast quantity of unabeled publically available texts. The texts are translated into tokens. Every word in a line is a seperate token. The LLM also has to predict which words will come next as to obey the rules of grammar. These models also inherit the inaccuracies and biases of the data which they are trained on. One prominent example is Open AI’s Chat Gpt.

3. Computer Vision → Computer Vision is the field allowing Computers to analyse, understand and interpret high level information from still images, sequence of images and videos.

4. Expert Systems. → An expert system is concerned with solving complex problems and emulating problem solving and decision-making of humans. An Expert system is divided into two parts, the Inference Engine and the Knowledge Base. The Inference Engine interacts with the Knowledge Base by applying it’s Inference Rules and arrives to a conclusion or deduce new information.

Analysing the effects of these technologies

According to Harman Mark Ai techniques are proving to be well suited to the Software Engineering workplace as they are based on pre-existing human intelligence, which is the archetype of a noisy, ill defined, competing, conflicting, connected, complex, interactive system.

I will also be using analytics from the latest 2023 Stack Overflow Survey.

According to the survey only 29.4% of all the respondents have never used any Artificial Intelligence tooling in the development process while 43.78% of all the respondents have used some form of AI tooling and the rest (25.46%) plan to use it soon. This number stays the same amongst professional developes with 44.17% currently using AI tooling, 25.88% who are planning to use it soon and 29.55% who will never use it. However, amongst the developers who are learning to code, more than half of them (54.87%) are using these tools, along with a further 27.97% who plan to use it soon and 17.74% have never used and will never use it. This statisitic also indicates that the next generation of coders will be very dependent on these technologies for their work. If any disruption to these technologies occur, for maintaince or otherwise, we might be in a lot of trouble. We also need to keep in mind that we currently do not have any idea how these technologies might impact the coding habit of the learners, whether that be for the positive or negative, however in a study Majeed Kazemitabaar, Justin Chow, Carl Ka To Ma, barbara J. Ericson, David Weintrop and Tovi Grossman found that AI code generators allowed novice programmers to perform better and faster with the absence of frustration. It also did not impact the manual performance of the users without the presence of the code generators. They also implied that the presence of these technologies might scaffold the learning process for beginners.

27.72% of all the respondents are very favourable towards the use of AI in the development process, while 48.56% favour it. Only around 16.5% are indifferent towards it, 4.02% are unsure and 2.76% are unfavourable and only a negligible % of developers (0.44%) are very unfavourable towards the use of AI in the development process. Amongs the professional developers, the statisitics remain mostly the same, but for the people learning to code, the developers who are favourable increases from 48.56% to 50.06%. Being unsure also increases by the same percentage. 32.81% of all respondents believe that AI tools help with increased productivity, 25.17% belive that it helps with speeding up the learning process and around 24.96% believe that these tools result in greater efficiency. 13.31% say that it results in improved accuracy in coding and 3.75% say that it results in improvied collaboration.

With accuracy of these tools, more respondents (5.46% vs 2.85%) highly disrust the tooling than highly trusting it. 39.3% somewhat trust it and 21.71% somewhat distrust it and the rest 30.68% neither trust it nor distrust these tools. The reason of this might be two fold. The reasons for more people distrusting than trusting it can be two fold. Tools like Chat GPT can often hallucinate code, meaning the code does not really work, and you will be left to your own devices to find the soltuion. Recently Surameery, Nigar M. Shafiq, and Mohammed Y. Shakor emphasized that such tools can be very useful in solving problems, however they combined with other toolings to get the full use out of them. In a study Neil Perry, Megha Srivastava, Deepak Kumar and Den Bonch found that participants using AI code generators were more prone to introducing bugs and other security vulnerabilities in their code base, but at the same time these same people also claimed the code writtten by them to be of a higher security. On the other hand people only using these assistants to write helper functions or adjusting the parameters were likely to provide more secure solutions.

With those currently using using these AI tools, 82.55% are using it to write code, 48.89% for debugging and getting help, 34.37% for documentation purposes, 30.1% for learning about a codebase. 23.87% for testing code, 13.52% for project planning and a minority of users (less than 10%) for Committing and Reviewing code, deployment and monitoring and collaboration with team mates. With those who are only interested in using these AI toolings, 23.72% are using it to write code, 40.66% for debugging and getting help, 50.24% for Documenting a codebase, 48.97% for learning about a codebase, 55.17% for testing code, 38.54% for project planning. The largest divide comes with the fact that in those who are interested in using these AI toolings, 49.51% want to use it for committing and reviewing code, 45.44% for deployment and monitoring and 29.88% for collaborating with team mates. This suggests that these users might be overestimating the ability of these AI technologies.

If we look at the development workflow most people already believe that with the use of AI tooling, their development workflow will be very or somewhat different across the entire spectrum of the process. They expect the very different part to be Documenting code closesly followed by learning about a codebase and testing code, while very important these only make up a few aspects of the duties of a software programmer. This indicates that these technologies might actually improve the development workflow rather than completely disrupting the software sector job market. The most somewhat different part of their workflow will be writing code, learning about a codebase, and debugging and getting help. They expect Deployment and monitoring and Collaborating with team mates to be very similar. This suggests that there will be still a need for software engineers but the work will shift to maintaince rather than directly writing code.

AI tools like Chat GPT can also be used to migrate and rewrite existing codebases in other languages or switch the framework entirely. In a study on AI code translations Justin D. Weisz, Michael Muller, Steven I. Ross. Fernando Martinez, Stephanie Houde, Mayank Agarwal, Kartik Talamapdupula and

John T. Richards found that the quality of code written by 59% of the professional software engineers improved when working with AI code generators. The participants felt that the AI tooling was useful in completing the translation task. The engineers who initally found it hard to translate code from java to python were able to do so with the help of these tools. Even though the participants had a perceived improvement in performance with the tooling, the time taken might be higher as they also had to spend time fixing the errors produced by these tools. This in turn shifted the nature of the task from writing to reviewing code.

It is also necessary to examine the way these technologies are trained. Github CoPilot and Code Whisperer are trained on the users using them and or the source code which is publically accessible through the use of services such as Github, Gitlab, Bit Bucket and or any other publically accessible resources. These lines of code are used without the authorization of the original code writer, which may consitiute a copyright violation, however in a study regarding Legal Risk Evaluation for Training LLMS on Copyrighted Texts Noorjahan Rahman and Eduardo Santacana concluded that the training of a LLM does not itself constitute a violation, however when the LLM explains the information or generates code for a user, it may inadvertently use copyrighted text.

At this point in time code generation using are widely being used as a productivity tool by both proffessional developers and novices. Code generators vastly increase the productivity of the user, sometimes at the cost of generating code which does not work or is very error prone. It may be used by new users to improve their speed of learning and using these generators may even translate into improvements when writing code manually. Using these generators for mission critical code may result in higher number of security vulnerabilties however they are very suitable for writing helper functions. There is only a small percentage of proffessional developers who are opposed to the use of these generators.

Implications and recommendations

The implications of continued innovation and use of AI code generators in the software development workflow are high. Newcomers to the field of Software development enjoy great advantage to the use of these technologies. It reduces some of the hassle associated with this field while also enjoying improvements to their own skill as developers. Proffessionals also greatly benefit however to a smaller degree. In the current form of these generators. They might be able to use them to write helper functions for the functioning of the codebase, however the use of these technologies in the current state with writing mission critical code might result in higher than usual number of security vulnerabilities. However with more advancements in the field, it might result in the entire process being automated. Current day novices and proffessional developers need to keep this in mind and improve other abilities which are are also part of their job description just as designing of the features of an application and the maintaince of a codebase. Developers should also use these technologies to make the process of migration of a codebase faster for themselves.

Conclusions

The roles present the software sector which are directly involved in writing code and debugging might be the first to be automated, however new jobs will instead be created to replace this role. The design specifications of a project still need to be worked out by teams of designers/engineers. After the code is written by the AI, more jobs are created for humans. They will need to test and make sure that there are not any logical errors present within the project. Further maintenance of the code base by monitoring the reviews and seeing the bug requests also needs to take place. Again the coding part will be automated, however seeing which changes from feedback need to integrated will still remain within the realm of human control. This conclusion assumes the fact that the large langugae models such as Chat-GPT and other tools like Github Co Pilot become advanced enough to write and debug code on their own. However this is certainly a possibility as a similar capability exists in GPT-4 with the use of plugins. However, if these AI code generators are not able to become more advanced, the job of a software engineer will become very automated but the need to manually verify the quality of the code will still exist.

Bibliography

Leavy, Susan, Barry O'Sullivan, and Eugenia Siapera. "Data, power and bias in artificial intelligence." *arXiv preprint arXiv:2008.07341* (2020).

Ford, Martin. "Could artificial intelligence create an unemployment crisis?." *Communications of the ACM* 56.7 (2013): 37-39.

Hamet, Pavel, and Johanne Tremblay. "Artificial intelligence in medicine." *Metabolism* 69 (2017): S36-S40.

Holzinger, Andreas, et al. "Causability and explainability of artificial intelligence in medicine." *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery* 9.4 (2019): e1312.

Ramesh, A. N., et al. "Artificial intelligence in medicine." *Annals of the Royal College of Surgeons of England* 86.5 (2004): 334.

Etzioni, Amitai, and Oren Etzioni. "Incorporating ethics into artificial intelligence." *The Journal of Ethics* 21 (2017): 403-418.

Wasilow, Sherry, and Joelle B. Thorpe. "Artificial intelligence, robotics, ethics, and the military: A Canadian perspective." *AI Magazine* 40.1 (2019): 37-48.

Bundy, Alan. "Preparing for the future of artificial intelligence." (2017): 285-287.

Tandon, Divya, Jyotika Rajawat, and Monisha Banerjee. "Present and future of artificial intelligence in dentistry." *Journal of Oral Biology and Craniofacial Research* 10.4 (2020): 391-396.

Alexander, Alan, et al. "An intelligent future for medical imaging: a market outlook on artificial intelligence for medical imaging." Journal of the American College of Radiology 17.1 (2020): 165-170.

Dhar, Vasant. "The future of artificial intelligence." *Big Data* 4.1 (2016): 5-9.

Nasteski, Vladimir. "An overview of the supervised machine learning methods." *Horizons. b* 4 (2017): 51-62.
Stack Overflow 2023 survey.

Mahesh, Batta. "Machine learning algorithms-a review." *International Journal of Science and Research (IJSR).[Internet]* 9.1 (2020): 381-386.

LeCun, Yann, Yoshua Bengio, and Geoffrey Hinton. "Deep learning." *nature* 521.7553 (2015): 436-444

Zohair, Lubna Mahmoud Abu. "The Future of Software Engineering by 2050s: Will AI Replace Software Engineers?." *International Journal of Information Technology* 2.3 (2018): 1-13.

Harman, Mark. "The role of artificial intelligence in software engineering." *2012 First International Workshop on Realizing AI Synergies in Software Engineering (RAISE)*. IEEE, 2012.

Latah, Majd, and Levent Toker. "Artificial intelligence enabled software-defined networking: a comprehensive overview." *IET networks* 8.2 (2019): 79-99.

Weisz, Justin D., et al. "Better together? an evaluation of ai-supported code translation." *27th International Conference on Intelligent User Interfaces*. 2022.

Becker, Brett A., et al. "Programming is hard-or at least it used to be: Educational opportunities and challenges of ai code generation." *Proceedings of the 54th ACM Technical Symposium on Computer Science Education V. 1*. 2023.

Surameery, Nigar M. Shafiq, and Mohammed Y. Shakor. "Use chat gpt to solve programming bugs." *International Journal of Information Technology & Computer Engineering (IJITC) ISSN: 2455-5290* 3.01 (2023): 17-22.

Kazemitabaar, Majeed, et al. "Studying the effect of AI Code Generators on Supporting Novice Learners in Introductory Programming." *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems*. 2023.

Perry, Neil, et al. "Do users write more insecure code with AI assistants?." *arXiv preprint arXiv:2211.03622* (2022).

Sorte, Bhagyashree W., Pooja P. Joshi, and Vandana Jagtap. "Use of artificial intelligence in software development life cycle—a state of the art review." *International Journal of Advanced Engineering and Global Technology* 3.3 (2015): 398-403.

Kulkarni, Rajesh H., and Palacholla Padmanabham. "Integration of artificial intelligence activities in software development processes and measuring effectiveness of integration." *Iet Software* 11.1 (2017): 18-26.