Инженерный вестник Дона, №5 (2022)
ivdon.ru/ru/magazine/archive/n5y2022/7675

Обзор вспомогательных инструментов на основе машинного обучения для написания исходного кода программ

Д.Ю. Какутин, А.С. Дмитриев, И.М.Абрамов

Волгоградский государственный технический университет

Аннотация: В последние годы набирает обороты использование различных средств на основе машинного обучения в процессе написания исходного кода различных программ, интерфейсов, веб-сайтов. К их числу можно отнести программы, помогающие в тестировании приложений, программы, анализирующие код разработчика, а также программы-ассистенты, помогающие писать код прямо в процессе, предсказывая и подсказывая разработчику варианты готового кода программы. В данной статье как раз будут рассмотрены подобные программы-ассистенты с целью анализа недостатков и обоснования необходимости развития функционала в данном направлении.
Ключевые слова: исходный код, машинное обучение, нейронные сети, проблема тестирования приложений, обработка естественного языка.

Подходы, использующие глубокий анализ данных и машинное обучение в тех или иных проявлениях в различных областях применяются очень давно [1], но в последний десяток лет случился небывалый до этого виток развития.

К существующим подходам добавились новые, современные, использующие современные алгоритмы, и, благодаря развитию также в сфере обработки и аналитики – большие объемы хорошо структурированных данных, с которыми можно было работать и совершенствовать старые подходы и алгоритмы [2].

Из популярных и часто используемых на данный момент нейронных сетей можно выделить:

1. Convolutional Neural Network (далее CNN) – свёрточная нейронная сеть, помогающая в задачах обработки графической информации, обнаружения и классификации объектов на изображениях. Более понятной аналогией для работы CNN является работа глаза человека – благодаря каким-либо

характеристикам видимого изображения мы классифицируем видимые глазом элементы, то же самое реализует CNN в основе своей логики [3].

2. Deep Belief Network (далее DBN) – сеть глубокого доверия, реализующая послойную структуру, в которой нейроны сети в одном слое связаны с нейронами в соседнем и при этом не связаны друг с другом. DBN используются в различных сценариях в реальной жизни, например, в медицине [4, 5].

3. Reccurent Neural Network (далее RNN) – рекуррентная нейронная сеть. Такая нейронная сеть состоит из узлов, соединенных друг с другом в определенной последовательности, ясно направленной. Применяются для распознавания рукописного текста или речи человека [6].

4. Long Short-Term Memory Network (далее LSTM) - Сеть с долговременной краткосрочной памятью, подвид RNN, который в целом можно выделить в отдельный пункт. Показывает лучшие результаты в задачах распознавания текста или же Обработки Естественного Языка - Natural Language Processing (далее NLP) [7, 8].

С задачами NLP так или иначе связано все, что касается текста, будь то рукописный текст, или же пользовательский ввод в любом приложении на компьютере. В том числе это, конечно же, касается исходного кода программ, так как большинство языков программирования используют в основе английский язык в сочетании с различными символьными операторами, поэтому в целом анализ исходного кода становится задачей NLP [9].

На данный момент уже существует множество инструментов для анализа пользовательского ввода при написании кода и предсказаний последующих участков кода и исправления ошибок.

Рассмотрим основные из них:

1. Github Copilot – совместная разработка компании Github INC (USA, California) и проекта OpenAI компании OpenAI INC (USA, California). Выпущен как плагин для IDE (Integrated development environment – интегрированная среда разработки) Visual Studio Code, позже выпущены версии для других различных IDE. Базируется на модели OpenAI Codex. Это языковая модель нейронной сети, направленная на генерацию рукописного текста. В случае задач с исходным кодом, Codex может предложить варианты решения (готовый код). Также нейросеть может преобразовывать ввод на английский язык и транслировать его между множеством решений для разных языков программирования [10]. Нейросеть была натренирована на выборках из английского языка, различного публично доступного исходного кода, а также с использованием множества репозиториев Github, включающих в себя около 54 миллионов публичных репозиториев на языке программирования Python. Функционал Copilot позволяет превращать комментарии с описанием в рабочий код, предлагать автозаполнение с готовым кодом для различных участков или отдельных функций или классов. При этом ассистент не всегда является надежным помощником. Для неопытного программиста автозаполянемый ассистентом код может стать ловушкой, ведь он не знает о различных проблемах в нем. К примеру, помимо прочего, существует возможность того, что предложенный ассистентом код будет устаревшим, неиспользуемым в обновленном окружении языка либо откровенно уязвимым. Это ставит под угрозу работоспособность разрабатываемых решений и, вполне можно считать это основным, главным минусом Copilot.

2. TabNine – летом 2019 года израильский стартап Codota выпустил бета-версию ассистента TabNine. Данный ассистент базируется на модели GPT-2 (устаревшая версия GPT-3, являющейся основной для OpenAI Codex). В целом TabNine предоставляет аналогичный Copilot функционал за

небольшими различиями. Во-первых, стоит упомянуть, что существует большая разница в исходных данных для тренировки нейросети, - у TabNine это около 2 миллионов файлов исходного кода против нескольких сотен миллионов у Copilot. С другой стороны, ассистент совершенствуется все время и его база знаний со временем растет, а точность модели повышается. Имеет функционал, позволяющий включать предсказание больших участков кода, это является одновременно и плюсом, и минусом, так как вероятность ошибок, неточностей, уязвимостей с количеством строк кода растет. Кроме прочего, является решением с платной версией, предоставляющей более широкий спектр возможностей по автозаполнению кода, в том числе использование подключения к серверам TabNine и новым знаний для модели предсказаний. В свою очередь, пользователь может отправлять данные своего исходного кода для тренировки и улучшения модели предсказаний, это снова же можно отнести и к плюсам, и к минусам. С одной стороны, с каждым днем точность модели хоть немного, но растет, и в долгосрочной перспективе это положительно скажется на работе ассистента для каждого пользователя. С другой – могут возникнуть ситуации, когда защищенный авторскими правами на интеллектуальную собственность код с помощью ассистента попадется кому-то в результате генерации.

3. IntelliCode – разработка компании Microsoft Corporation (USA, Washington), идейный продолжатель инструмента IntelliSense – справочника с автозаполнением кода, работающим в IDE компании. На этот раз в дополнении к справочнику используется нейронная сеть. В отличии от своих собратьев, IntelliCode делает больший упор на качество и точность автозаполняемого кода, длина предсказываемых ассистентом автозаполнений отличается от Copilot и TabNine в меньшую сторону, хотя, возможно, в будущем это изменится. При этом IntelliCode имеет гораздо меньший спектр поддерживаемых языков программирования и еще меньший спектр

поддерживаемых IDE для использования. Как видим, в угоду качеству в жертву приносятся количественные характеристики и поддержка различного функционала.

В целом, кроме описанных инструментов существует еще больше десятка представителей данного рода приложений, некоторые из них, возможно, заслуживают упоминания, но представляют совершенно аналогичный функционал, при этом качественно отличающийся в худшую сторону из-за различных факторов, в основном, конечно же изначальные разработки, в том числе модель нейронной сети, а также наличие объемного набора данных исходного кода и больших вычислительных мощностей для тренировки нейронной сети. Таким образом, на основании вышеизложенного, можно отметить, что текущим программам, направленным на ассистирование в написании кода требуются дополнительные функциональные возможности для выявления уязвимого, устаревшего кода, а также тестирования уже сгенерированного кода. Это позволит в общем повысить точность и надежность ассистируемых разработок и уменьшить затраты времени и денег на отладку приложений.

Литература

1. Горбачевская Е.Н., Краснов С.С. История развития нейронных сетей. // Вестник Волжского университета. 2015, № 1 (23). URL: cyberleninka.ru/article/n/istoriya-razvitiya-neyronnyh-setey.

2. Galván Edgar, Mooney Peter. Neuroevolution in Deep Neural Networks: Current Trends and Future Challenges // IEEE Transactions on Artificial Intelligence. 2021. Volume 2, pp. 476-493, DOI: 10.1109/TAI.2021.3067574

3. Indola Sakshi, Kumar Goswami Anil, Mishrab S.P., Asopa Pooja. Conceptual Understanding of Convolutional Neural Network- A Deep Learning Approach. // Procedia Computer Science. 2018. Volume 132, pp. 679-688, DOI: 10.1016/J.PROCS.2018.05.069.

4. Movahedi F., Coyle J.L., Sejdić E. Deep belief networks for electroencephalography: A review of recent contributions and future outlooks. // IEEE J Biomed Health Inform. 2018. Volume 22, pp. 642-652, DOI: 10.1109/JBHI.2017.2727218.

5. Стебаков И.Н., Шутин Д.В., Марахин Н.А. Машинное обучение в реабилитационной медицине и пример классификатора движений пальцев для кистевого тренажера // Инженерный вестник Дона, 2020, №6. URL: ivdon.ru/ru/magazine/archive/n6y2020/6514.

6. Sherstinsky Alex. Fundamentals of Recurrent Neural Network (RNN) and Long Short-Term Memory (LSTM) network // Physica D: Nonlinear Phenomena. 2020. Volume 404, DOI: 10.1016 /j.physd.2019.132306.

7. Yao Lirong, Guan Yazhuo. An Improved LSTM Structure for Natural Language Processing // 2018 IEEE International Conference of Safety Produce Informatization (IICSPI), DOI: 10.1109/IICSPI.2018.8690387.

8. Богомолов Ю.А. Обзор моделей нейронных сетей для обработки естественного языка // StudNet. 2020, №4, с. 203–217.

9. Mokhov Serguei, Paquet Joey, Debbabi Mourad. The Use of NLP Techniques in Static Code Analysis to Detect Weaknesses and Vulnerabilities // Conference: The 27th Canadian Conference on Artificial Intelligence. 2014, pp 326-332, DOI: 10.1007/978-3-319-06483-3_33.

10. Finnie-Ansley James, Denny Paul, A. Becker Brett, Luxton-Reilly Andrew, Prather James. The Robots Are Coming: Exploring the Implications of OpenAI Codex on Introductory Programming. // ACE '22: Australasian Computing Education Conference. 2022, pp. 10-19, DOI: 10.1145/3511861.3511863.

References

1. Gorbachovskaya E.N., Krasnov S.S. Vestnik Volzhskogo universiteta. 2015, № 1 (23). URL: cyberleninka.ru/article/n/istoriya-razvitiya-neyronnyh-setey.

2. Galván Edgar, Mooney Peter. IEEE Transactions on Artificial Intelligence. 2021. Volume 2, pp. 476-493, DOI: 10.1109/TAI.2021.3067574

3. Indola Sakshi, Kumar Goswami Anil, Mishrab S.P., Asopa Pooja. Procedia Computer Science. 2018. Volume 132, pp. 679-688, DOI: 10.1016/J.PROCS.2018.05.069.

4. Movahedi F., Coyle J.L., Sejdić E. Deep belief networks for electroencephalography: IEEE J Biomed Health Inform. 2018. Volume 22, pp. 642-652, DOI: 10.1109/JBHI.2017.2727218.

5. Stebakov I.N., Shutin D.V., Marahin N.A. Inzhenernyj vestnik Dona, 2020, №6. URL: ivdon.ru/ru/magazine/archive/n6y2020/6514.

6. Sherstinsky Alex. Physica D: Nonlinear Phenomena. 2020. Volume 404, DOI: 10.1016 /j.physd.2019.132306.

7. Yao Lirong, Guan Yazhuo. 2018 IEEE International Conference of Safety Produce Informatization (IICSPI), DOI: 10.1109/IICSPI.2018.8690387.

8. Bogomolov Y.A. StudNet. 2020, №4, pp 203-217.

9. Mokhov Serguei, Paquet Joey, Debbabi Mourad. Conference: The 27th Canadian Conference on Artificial Intelligence. 2014, pp 326-332, DOI: 10.1007/978-3-319-06483-3_33.

10. Finnie-Ansley James, Denny Paul, A. Becker Brett, Luxton-Reilly Andrew, Prather James. ACE '22: Australasian Computing Education Conference. 2022, pp. 10-19, DOI: 10.1145/3511861.3511863.