Трансляция нейросетевых моделей в программный код на языке программирования высокого уровня

С.А. Ямашкин[1], Е.О. Ямашкина[1,2]

[1]Мордовский государственный университет им. Н. П. Огарева, Саранск
[2]Российский технологический университет (МИРЭА), Москва

Аннотация: В работе представлен принцип работы алгоритма трансляции графического представления нейросетевых моделей в программный код в рамках репозитория нейронных сетей. На основе полученной структуры данных можно осуществить последовательную генерацию программного кода на языке программирования общего назначения.
Ключевые слова: нейросетевая модель, нейронная сеть, репозиторий, графовая модель, программирование, трансляция, пространственные данные, алгоритм, топология, архитектура.

1. Введение

Использование нейросетевых моделей в решении практических проектных задач в области анализа структуры земель и прогнозирования развития природных и природно-техногенных процессов может быть осуществлено при условии консолидации накапливаемого реестра моделей машинного обучения в единый репозиторий, для взаимодействия с которой организованы графические пользовательские и прикладные программные интерфейсы [1]. Практическую ценность репозиторий приобретает при достижении возможности трансляции графического представления моделей в программный код. Разработка формализованной схемы хранения моделей глубокого машинного анализа пространственных данных в форме метаязыка позволяет осуществить возможность их конвертации в представления, используемые современными фреймворками машинного обучения.

2. Методология и методы исследования

Для решения задачи трансляции графического представления нейросетевой модели в программный код, ее структура должна быть описана ориентированным ациклическим графом, орграфом без направленных

циклов, но возможным наличием параллельных путей из одного узла в другой. Вершины графа при этом представляют собой слои или блоки, а ребра – связи и ограничения, согласно которым определяется топологическая структура модели. Ключевым алгоритмом преобразования графического представления модели в инструкции на языке программирования общего назначения может стать топологическая сортировка, представляющая собой линейное упорядочивание вершин ориентированного ациклического графа согласно частичному порядку, заданному ребрами этого на множестве его вершин, такое, что для каждого направленного ребра вершины-источника к конечной вершине, вершина источника становится предшествующей [2 – 4].

Классические алгоритмы топологической сортировки имеют линейную асимптотическую сложность, оцениваемую как сумма количества узлов графа и его ребер: $\overline{O(|V|+|E|)}$. Для выполнения алгоритма необходимо сформировать графовую структуру G на основе граф-схемы модели *GRAPH*. Для этого в нее необходимо перенести множество вершин-слоев *LAYERS* и связей *LINKS*. После этого в графе нейросетевой модели G необходимо найти список стартовых узлов, не имеющих входящих связей, и добавить их в множество S. Затем необходимо осуществить итерационное перемещение вершин-слоев из множества S в множество L. Для перемещенной вершины n осуществляется выбор всех вершин-слоев m, для которых узел n является предшествующим на основе связи e. По выполнении этой операции связь e удаляется из графа G. Если при этом вершина-слой m не имеет входящих связей, ее необходимо добавить в множество S. Обозначенную последовательность шагов необходимо выполнять до тех пор, пока множество S не станет пустым. Если при этом в графе G не осталось связей, то алгоритм можно считать выполненным успешно, при этом структура данных L будет содержать топологически упорядоченное множество вершин.

В противном случае, можно говорить о том, что исходный граф содержит циклы, что приводит к невозможности трансляции графической модели.

3. Результаты исследования

Достичь асимптотической сложности $O(|V|)$ можно за счет модификации алгоритма с использованием стратегии поиска в глубину с добавлением слоя-вершины в момент выхода из неё с применением рекурсии. Алгоритм перебирает слои-вершины графа G в произвольном порядке, инициируя поиск в глубину, который завершается при достижении терминального слоя-вершины или вершины, которая уже был посещена алгоритмом ранее при выполнении топологической сортировки. На верхней уровне декомпозиции алгоритм топологической сортировки блоков нейросетевой модели на основе стратегии поиска в глубину обрабатывает вершины-слои графа *G*, которым по умолчанию не сопоставлен ни один тип меток (временных или постоянных).

Алгоритм посещения вершины-слоя, представленный на рис.1, включает несколько этапов. Сперва осуществляется проверка наличия у слоя-вершины меток: в случае наличия постоянной метки посещение вершины завершается, а в случае наличия временной метки – завершается выполнение всего алгоритма с выводом о наличии циклов в анализируемом графе нейросетевой модели. Затем посещаемой вершине сопоставляется временная метка и выбираются все последующие за счет связей вершины-слои. Для каждой из них, в свою очередь, рекурсивно запускается алгоритм посещения вершины. Наконец, после обхода всех последующих вершин, посещаемой вершина-слой добавляется в начало топологически упорядоченного множества вершин *L*, для нее удаляется временная метка и добавляется постоянная. Таким образом, обработка вершин-слоев графа без постоянных

веток в итоге приведет к формированию типологически отсортированного списка слоев нейросетевой модели.

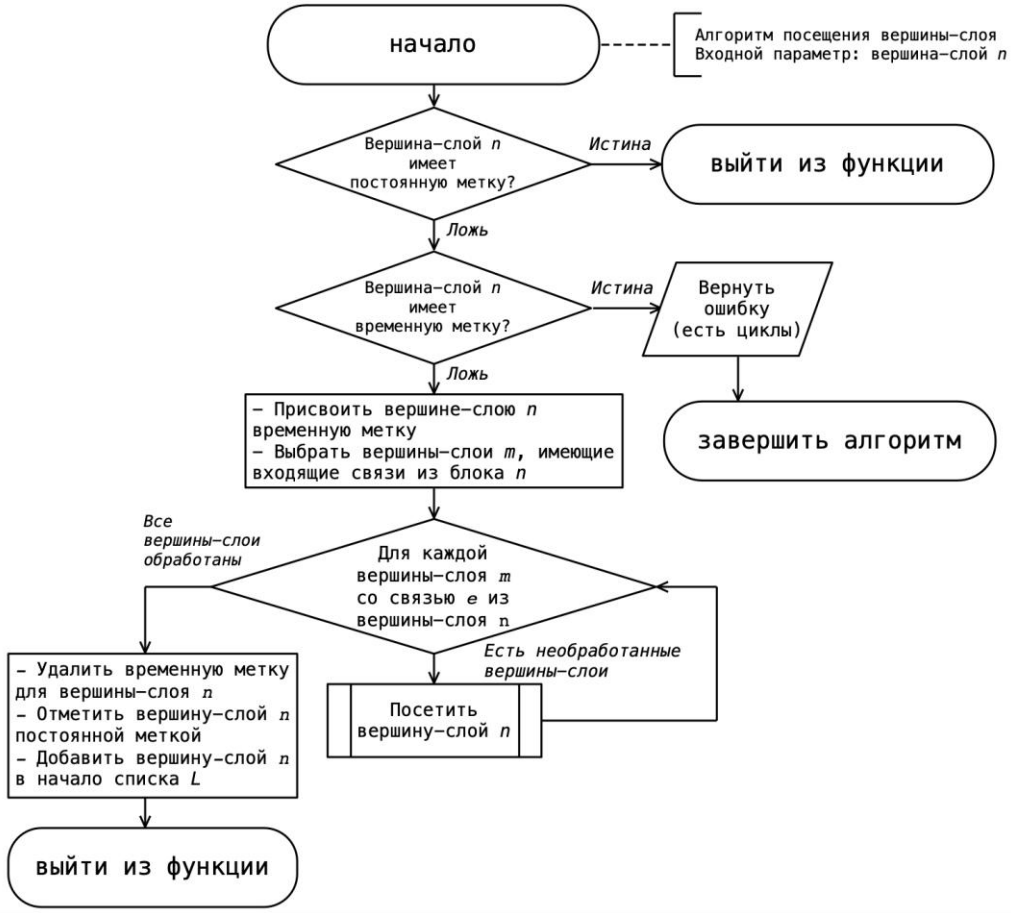

Рис. 1. – Рекурсивный алгоритм топологической сортировки слоев-вершин нейросетевой модели на основе стратегии поиска в глубину

На основе полученной структуры данных при использовании информации о связях в графе нейросетевой модели можно осуществить последовательную генерацию программного кода на языке программирования общего назначения. В рамках репозитория нейросетевых моделей для анализа пространственных данных реализована трансляция графического представления моделей в представление на языке Python, основанное на применении каркаса глубокого машинного обучения Keras, предоставляющего гибкие прикладные программные интерфейсы (API) для построения нейросетевых моделей поверх платформы машинного обучения TensorFlow [5,6]. Интерес представляют функциональные программные

интерфейсы Keras, позволяющие организовывать гибкие модели с ветвящейся нелинейной топологией, обобщенными блоками и множеством входов и выходов. Возможности функциональных программных интерфейсов в полной мере удовлетворяют требованию информативного отображения предложенной графовой структуры нейросетевой модели репозитория в программный код на языке программирования [7]. Для трансляции графовой модели в программный код необходимо сформировать следующие программные блоки:

1) Подключение модулей каркаса исходя из слоев-вершин графовой модели нейронной сети; определение настроечных переменных и их значений с целью их дальнейшего использования в рамках глубокой нейросетевой модели [8]. В случае использования пользовательских слоев и блоков на их основе, не определенных в Keras, обеспечивается возможность импорта необходимого программного кода посредством подключения нужного компонента.

2) Вывод топологически отсортированных инструкций, направленных на генерацию структуры нейронной сети на основе сформированной графовой модели; группировка слоев нейросетевой модели в объект с функциями обучения и вывода.

3) Компиляция модели с определением параметрически заданной функции оптимизации, оценки потерь, оценочных метрик и других параметров обучения; описание инструкции по обучению модели в течение фиксированного количества эпох, на основе определенных параметров валидации и других настроек. Формирование перечня параметров компиляции и обучения модели в нотации языка Python производится на основе данных промежуточной формализованной схемы хранения нейросетевых моделей в формате XML или JSON.

4. Выводы

Использование репозитория глубоких нейросетевых моделей позволяет подойти к решению научной проблемы интеграции нейронных сетей предварительно обученных моделей с возможностью их последующего использования для решения проектных задач цифровой экономики [9,10]. В рамках репозитория нейросетевых моделей реализована трансляция графического представления модели в код на языке MATLAB. Алгоритм работы транслятора в этом случае имеет схожую структуру с тем исключением, что в нотации языка MATLAB и пакета прикладных модулей Deep Learning Toolbox знания об узлах и связях графа нейросетевой модели хранятся в отдельных массивах.

5. Благодарности

Работа выполнена при финансовой поддержке гранта Президента Российской Федерации (грант № МК-199.2021.1.6).

Литература

1. Ямашкин, С. А., Ямашкин А. А., Занозин В. В. Формирование репозитория глубоких нейронных сетей в системе цифровой инфраструктуры пространственных данных // Материалы IX Международного научного форума молодых ученых, инноваторов, студентов и школьников, Астрахань, 28–29 апреля 2020 года, 2020. С. 370-375.

2. Kahn A. B. Topological sorting of large networks Communications of the ACM. 1962. Vol. 5. № 11. pp. 558-562.

3. Full Grammar specification. URL: docs.python.org/3/reference/grammar.html.

4. Han J., Haihong E., Le G., Du J. Survey on NoSQL database. 2011 6th international conference on pervasive computing and applications. 2011. pp. 363-366.

5. Bengio Y. Learning deep architectures for AI // Foundations and Trends in Machine Learning. 2009. vol. 2. № 1. pp. 1-127.

6. Шолле Ф. Глубокое обучение на Python. – СПб.: Питер, 2018. 400 с.

7. Галушка В. В., Фатхи В. А. Формирование обучающей выборки при использовании искусственных нейронных сетей в задачах поиска ошибок баз данных // Инженерный вестник Дона. 2013, № 2. URL: ivdon.ru/ru/magazine/archive/n2y2013/1597.

8. Соловьев, Р. А., Тельпухов Д. В., Кустов А. Г. Автоматическая сегментация спутниковых снимков на базе модифицированной свёрточной нейронной сети UNET // Инженерный вестник Дона. 2017, № 4. URL: ivdon.ru/ru/magazine/archive/n4y2017/4433.

9. Ямашкин С. А., Ямашкин А. А. Анализ производительности и оптимизация высоконагруженных геопортальных систем // Современные наукоемкие технологии. 2021. № 10. С. 108-112.

10. Ямашкин С. А., Ямашкин А. А., Ямашкина Е. О., Занозин В. В. Интеграция знаний в цифровых инфраструктурах пространственных данных // Саранск: Национальный исследовательский Мордовский государственный университет им. Н.П. Огарёва, 2021. 216 с.

References

1. Yamashkin, S. A., Yamashkin A. A., Zanozin V. V. Materialy IX Mezhdunarodnogo nauchnogo foruma molodyh uchenyh, innovatorov, studentov i shkol'nikov, Astrahan', 28–29 aprelja 2020 goda, 2020. pp. 370-375.

Kahn A. B. Topological sorting of large networks Communications of the ACM. 1962. Vol. 5. № 11. pp. 558-562.

3. Full Grammar specification. URL: docs.python.org/3/reference/ grammar.html.

4. Han J., Haihong E., Le G., Du J. Survey on NoSQL database. 2011 6th international conference on pervasive computing and applications. 2011. pp. 363-366.

5. Bengio Y. Foundations and Trends in Machine Learning. 2009. vol. 2. № 1. pp. 1-127.

6. Sholle F. Glubokoe obuchenie na Python. [Deep Learning in Python]. SPb: Piter, 2018. 400 p.

7. Galushka V. V., Fathi V. A. Inzhenernyj vestnik Dona, 2013. № 2. URL: ivdon.ru/ru/magazine/archive/n2y2013/1597.

8. Solov'ev, R. A., Tel'puhov D. V., Kustov A. G. Inzhenernyj vestnik Dona, 2017. № 4. URL: ivdon.ru/ru/magazine/archive/n4y2017/4433.

9. Yamashkin S. A., Yamashkin A. A. Sovremennye naukoemkie tehnologii. 2021. № 10. pp. 108-112.

10. Yamashkin S. A., Yamashkin A. A., Yamashkina E. O., Zanozin V. V. Saransk: Nacional'nyj issledovatel'skij Mordovskij gosudarstvennyj universitet im. N.P. Ogarjova, 2021. 216 p.