УДК 004.032.26 https://doi.org/10.26160/2541-8637-2025-15-89-92

ПОВЫШЕНИЕ КАЧЕСТВА РАЗРАБОТКИ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ ДЛЯ ТЕХНИЧЕСКИХ ОБЪЕКТОВ МАШИНОСТРОЕНИЯ НА ОСНОВЕ НЕЙРОННЫХ СУРРОГАТОВ

Ерохин В.В.[1,2], Зафиров А.Е.[2]
[1]Московский государственный институт международных отношений (университет) Министерства иностранных дел РФ, Москва;
[2]Московский государственный технический университет имени Н.Э. Баумана, Москва

Ключевые слова: искусственные нейронные сети, машинное обучение, суррогатные модели, суррогатная компиляция, суррогатная адаптация, суррогатная оптимизация.
Аннотация. В статье формализуется таксономия шаблонов проектирования на основе нейронных суррогатных программных моделей. Описывается методология программирования относительно общей для всех шаблонов проектирования программ. Исследование закладывает фундамент для нового класса рабочих процессов инженеров-программистов, основанных на программировании с использованием суррогатов программ.

IMPROVING THE QUALITY OF SOFTWARE DEVELOPMENT FOR TECHNICAL OBJECTS OF MECHANICAL ENGINEERING BASED ON NEURAL SURROGATES

Erokhin V.V.[1,2], Zafirov A.E.[2]
[1]Moscow state institute of international relations (university), Ministry of Foreign Affairs of the Russian Federation, Moscow;
[2]Bauman Moscow State Technical University, Moscow

Keywords: artificial neural networks, machine learning, surrogate models, surrogate compilation, surrogate adaptation, surrogate optimization.
Abstract. The article formalizes a taxonomy of design patterns based on neural surrogate software models. A programming methodology is described that is relatively common to all software design patterns. The research lays the foundation for a new class of software engineers' workflows based on programming using software surrogates.

Концепция программирования с использованием суррогатных программ на основе искусственных нейронных сетей (programming with neural surrogates of programs) сочетает элементы программирования и машинного обучения с использованием нейронных сетей для создания суррогатных программных моделей для электронно-вычислительной техники (ЭВТ). Эти суррогатные модели являются приближенными представлениями оригинальных программ или систем и могут использоваться для упрощения процесса разработки, оптимизации и анализа программного обеспечения в ИТ-отрасли.

Использование нейронных суррогатов в разработке программного обеспечения для технических объектов машиностроения может значительно ускорить этот процесс, особенно в тех случаях, когда ресурсоемкие симуляции или сложные вычислительные задачи требуют значительных затрат времени и

вычислительных мощностей. Нейронные суррогаты позволяют тестировать и прорабатывать идеи быстрее, снижая время на итерации и исследования.

Типичные примеры программ-суррогатов для ЭВТ включают нейронные сети [1], гауссовские процессы, линейные модели [2] и случайные леса [3]. Из этих архитектур моделей нейронные суррогаты стали популярным вариантом замещения классического программирования [1], поскольку для многих задач нейронные сети представляют собой современные модели, обеспечивающие высокую точность и быстроту выдачи требуемого результата [4].

По сравнению со стандартными рабочими процессами разработки программ для ЭВТ, программирование с использованием суррогатов требует более низких затрат на разработку [1] и приводит к программам с более низкой стоимостью их проектирования или более высоким качеством результата [1]. Однако представленные в научной литературе подходы как к применению, так и к разработке суррогатов разрозненны, без единой таксономии или методологии разработки.

Концептуально суррогатное программирование должна как минимум содержать три рабочих процесса проектирования программ для ЭВТ: суррогатная компиляция, суррогатная адаптация и суррогатная оптимизация.

Суррогатная компиляция позволяет программистам разрабатывать суррогат, который копирует поведение программы для развертывания конечным пользователям. Ключевые преимущества этого подхода включают возможность выполнения суррогата на другом оборудовании и возможность ограничить или ускорить время выполнения суррогата.

Параметры суррогатного программирования в данном исследовании сравниваются с LLVM-MCA (LLVM Machine Code Analyzer). LLVM-MCA – это инструмент, входящий в состав компилятора LLVM, который предназначен для анализа и оценки производительности машинного кода. Он предназначен для разработчиков и исследователей, стремящихся оптимизировать программное обеспечение на уровне низкого уровня, особенно в контексте архитектуры процессоров и их особенностей.

LLVM-MCA является эффективным инструментом для понимания внутренних процессов выполнения программ для ЭВТ. Он помогает разработчикам создавать более эффективный и производительный код, отвечающий требованиям современной ЭВТ.

Для проектирования программ необходимо обучить нейронную сеть повторению предсказания времени выполнения для данного фрагмента входного кода. Полученная в результате нейронная сеть работает в 1,75 раза быстрее, чем LLVM-MCA на том же оборудовании, с отклонением менее чем на 8% от прогнозов LLVM-MCA.

Суррогатная адаптация включает две фазы: программисты сначала разрабатывают суррогат программы; затем программисты дополнительно обучают этот суррогат на данных из другой задачи. Главные преимущества этого подхода включают в себя то, что суррогатная адаптация позволяет изменять семантику программы для выполнения другой интересующей задачи и что это

может быть более эффективным с точки зрения данных или приводить к более высокой точности, чем обучение модели с нуля для выполнения задачи.

На этом этапе обучаем нейронную сеть повторять предсказания LLVM-MCA, а затем точно настраиваем эту сеть на основе измерений синхронизации кода на физическом процессоре. При этом ошибка LLVM-MCA в этой нейронной сети при прогнозировании достоверных таймингов должна быть не более 5%.

Суррогатная оптимизация обусловливается разработкой программистами суррогата программы, где оптимизируются входные параметры этого суррогата, затем подключают оптимизированные параметры обратно к исходной программе. Главным преимуществом этого подхода является то, что суррогатная оптимизация может оптимизировать входные данные быстрее, чем оптимизация входных данных непосредственно для программы, из-за потенциальной более высокой скорости выполнения суррогата и возможности того, что суррогат может быть дифференцируемым, даже когда исходная программа таковой не является (с учетом оптимизации входных данных на основе использования метода градиентного спуска) [1].

На этом этапе происходит обучение нейронной сети по параметру "воспроизводить прогноз LLVM-MCA", при этом LLVM-MCA параметризуется различными наборами параметров моделирования. Далее идет оптимизация искусственной нейронной сети для того, чтобы найти параметры, которые позволяют нейронной сети точно прогнозировать истинное время выполнения кода программы. Затем необходимо вставить эти параметры в LLVM-MCA, что и позволит повысить точность LLVM-MCA минимум на 5% по сравнению с параметрами, выбранными экспертом.

Суррогатная оптимизация оптимизирует входные данные быстрее, чем оптимизация входных данных непосредственно по программе, из-за потенциально более высокой скорости выполнения суррогата и возможности дифференцирования суррогата, даже если исходная программа таковой не является [1].

В качестве архитектуры нейронной сети в суррогатном программировании в основном выбираются многослойные персептроны. Создав архитектуру нейронной сети, программист должен определить, как обучать нейронный суррогат.

Данные обучения суррогата определяют распределение входных данных, на которых суррогат, как ожидается, будет хорошо работать. Данные должны быть репрезентативными для входных данных для задачи ниже по потоку, для которой развернут суррогат. Данные также должны быть многочисленными и достаточно разнообразными, чтобы обучить суррогатную модель для обобщения наблюдаемого поведения программы.

Функция потерь – это цель в процессе оптимизации нейронной сети, которая измеряет, насколько плохо предсказание нейронной сети по сравнению с основной истиной.

Имея в распоряжении обучающие данные и функцию потерь, программист должен затем обучить суррогат. Это приводит к компромиссу между точностью и стоимостью обучения. Поскольку процедура обучения может выполняться

несколько раз во время поиска по гиперпараметру, пороговому значению или бюджетным затратам, которые должны быть установлены так, чтобы учесть полную стоимость проектирования и обучения нейронной сети.

В литературе существует два основных подхода к определению подходящего времени обучения суррогатной матери. Одним из подходов является обучение в течение фиксированного времени обучения, обычно определяемого с помощью экспериментов на проверочном наборе [1]. Другой подход заключается в обучении до достижения приемлемой точности, будь то через совокупность потерь в обучении или через достижение минимально приемлемой точности.

Вывод. Данное исследование демонстрирует перспективность использования суррогатов для разработки сложных систем, особенно в условиях, когда программистам не хватает полной характеристики системы и ее операционной среды.

Список литературы

1. Renda A., Chen Y., Mendis C., Carbin M. DiffTune: Optimizing CPU simulator parameters with learned differentiable surrogates. // 53rd Annual IEEE/ACM international symposium on microarchitecture (MICRO), Athens, Greece. 2020, pp. 442-455. DOI: 10.1109/MICRO50266.2020.00045.
2. Ding Y., Pervaiz A., Carbin M., Hofmann H. Generalizable and interpretable learning for configuration extrapolation // Proceedings of the 29th ACM joint meeting on european software engineering conference and symposium on the foundations of software engineering (ESEC/FSE 2021). Association for computing machinery, New York, NY, USA, pp. 728-740. doi.org/10.1145/3468264.3468603.
3. Nardi L., Souza A., Koeplinger D., Olukotun K. HyperMapper: a practical design space exploration framework // 2019 IEEE 27th international symposium on modeling, analysis, and simulation of computer and telecommunication systems (MASCOTS), Rennes, France. 2019, pp. 425-426. DOI: 10.1109/MASCOTS.2019.00053.
4. Devlin J., Chang M.-W., Lee K., Toutanova K. BERT: Pre-training of deep bidirectional transformers for language understanding // Proceedings of the 2019 conference of the north American chapter of the association for computational linguistics: human language technologies. 2019, vol. 1, pp. 4171-4186. doi.org/10.18653/v1/N19-1423.

Сведения об авторах:

Ерохин Виктор Викторович – д.т.н., доцент, профессор кафедры «Математические методы и бизнес-информатика» МГИМО, профессор кафедры «Инновационное предпринимательство» МГТУ им. Н.Э. Баумана;

Зафиров Александр Евгеньевич – магистрант.