ДОВЕРЕННОЕ АВТОМАТИЧЕСКОЕ МАШИННОЕ ОБУЧЕНИЕ ПРИ ФУНКЦИОНИРОВАНИИ ЦИФРОВЫХ ДВОЙНИКОВ

DOI: 10.36724/2072-8735-2024-18-7-44-55

Беззатеев Сергей Валентинович,
Санкт-Петербургский государственный университет аэрокосмического приборостроения, г. Санкт-Петербург, Россия, ***bsv@aanet.ru***

Фомичева Светлана Григорьевна,
Санкт-Петербургский государственный университет аэрокосмического приборостроения, г. Санкт-Петербург, Россия, ***levikha@mail.ru***

Жемелев Георгий Алексеевич,
Санкт-Петербургский политехнический университет Петра Великого, г. Санкт-Петербург, Россия, ***wws.dev@gmail.com***

Manuscript received 14 June 2024;
Accepted 10 July 2024

Ключевые слова: *цифровые двойники, автоматическое машинное обучение, доверенные вычислительные среды, оптимизация вычислительных ресурсов*

В настоящее время одной из основных задач цифровых двойников (Digital Twins) является не только детализованная симуляция промышленного объекта в соответствии с бизнес-целями, но и прототипирование сценариев прогнозной производственной деятельности. Цифровые двойники в ходе своей эксплуатации сопровождаются формированием датасетов колоссальных размеров, при этом адаптивное поведение цифровых двойников требует автоматической очистки сырых данных, выявления релевантных признаков датасетов и, при необходимости, трансферного обучения в режиме реального времени, а также конфиденциальности проводимого обучения. Обладая миллионами параметров и сотнями гиперпараметров, цифровые двойники в своем жизненном цикле все чаще опираются на AutoML – автоматические способы машинного обучения при оптимизации гиперпараметров, поиск эффективных конвейеров и архитектур ML-моделей, а также настройки динамических алгоритмов обучения. Выявленные в результате AutoML конфигурации ML-конвейеров и оптимизационные решения часто становятся активами компаний, являясь чувствительной информацией, которая требует при использовании облачных ресурсов дополнительной защиты не только на этапах хранения и передачи, но и на этапе обработки в силу необходимости их перманентной адаптации. В статье рассматривается задача оптимизации вычислительных ресурсов и механизмов сопряжения моделей автоматического машинного обучения и доверенных вычислительных сред при обновлении конфигураций конвейеров ML-моделей, поддерживающих функционирование компонентов цифрового двойника. Формализована задача и общий протокол доверенного автоматического машинного обучения. Приведены результаты анализа современных подходов к реализации конфиденциальных вычислений и распределенного автоматического машинного обучения, выявлены их особенности, достоинства и недостатки. Приведены результаты моделирования конфиденциального распределенного обучения для решения задачи многомерной нелинейной регрессии. Показано, что при автоматическом решении задач нелинейной многомерной регрессии использование методов автоматического распределенного обучения, позволяет добиться существенного повышения эффективности обработки данных и обеспечить конфиденциальность обработки гиперпараметров ML-конвейеров.

Информация об авторах:
Беззатеев Сергей Валентинович, *д.т.н., доцент, заведующий кафедрой информационной безопасности Санкт-Петербургского государственного университета аэрокосмического приборостроения, г. Санкт-Петербург, Россия*
Фомичева Светлана Григорьевна, *к.т.н., профессор, профессор Санкт-Петербургского государственного университета аэрокосмического приборостроения, г. Санкт-Петербург, Россия*
Жемелев Георгий Алексеевич, *аспирант Санкт-Петербургского политехнического университета Петра Великого, г. Санкт-Петербург, Россия*

Для цитирования:
Беззатеев С.В., Фомичева С.Г., Жемелев Г.А. Доверенное автоматическое машинное обучение при функционировании цифровых двойников // T-Comm: Телекоммуникации и транспорт. 2024. Том 18. №7. С. 44-55.

For citation:
Bezzateev S.V., Fomicheva S.G., Zhemelev G.A. Trusted automatic machine learning in the operation of digital twins. *T-Comm*, vol. 18, no. 7, pp. 44-55. (*in Russian*)

Введение

Технологии искусственного интеллекта (AI) де-факто стали широко применяться в производственной деятельности многими компаниями. По данным аналитического отчета за 2023 Global Trends in AI [1], компании, являющиеся флагманами по интеграции AI в свои производственные процессы, используют гибридный подход при развертывании рабочих AI-нагрузок, совмещая их размещение в корпоративных центрах данных и общедоступных облачных ресурсах. По состоянию на 2023 год облако является основным местом развертывания на этапах обучении моделей машинного обучения (ML-моделей) у 47% компаний, а на этапе эксплуатации ML-моделей - у 44% [1]. Причем тренд смещается в сторону предпочтения облачных ресурсов. Компании, являющиеся владельцами цифровых двойников, входят в когорту флагманов по интеграции ML-моделей на всех этапах их жизненного цикла.

В настоящее время одной из основных задач цифровых двойников (Digital Twins) является не только детализованная симуляция промышленного объекта в соответствии с бизнес-целями, но и прототипирование сценариев прогнозной производственной деятельности. Цифровые двойники в ходе своей эксплуатации сопровождаются формированием датасетов колоссальных размеров [2,3]. При этом адаптивное поведение цифровых двойников требует, с одной стороны, автоматической очистки сырых данных, выявления релевантных признаков датасетов и, при необходимости, трансферного обучения в режиме реального времени, а с другой стороны – конфиденциальности проводимого обучения. Обладая миллионами параметров и сотнями гиперпараметров, цифровые двойники в своем жизненном цикле все чаще опираются на AutoML – автоматические способы машинного обучения при оптимизации гиперпараметров, поиск эффективных конвейеров и архитектур ML-моделей, а также настройки динамических алгоритмов обучения [4, 5]. Выявленные в результате AutoML конфигурации ML-конвейеров и оптимизационные решения часто становятся активами компаний, являясь чувствительной информацией, которая требует при использовании облачных ресурсов дополнительной защиты не только на этапах хранения и передачи, но и на этапе обработки в силу необходимости их перманентной адаптации.

В данной статье рассматриваются возможности сбалансированного решения дилеммы между скоростью обучения и защитой гиперпараметров ML-моделей за счет использования автоматических методов машинного обучения (AutoML)[6] в доверенных вычислительных средах (Trusted Execution Environment – TEE) [7]. Актуальность разрешения данной дилеммы заключается в том, что на практике автоматическая оптимизация гиперпараметров (Hyper-Parameters Optimization - HPO) при машинном обучении цифровых двойников сталкивается с целым рядом проблем, которые делают его сложной задачей:

1) Большая вычислительная база AutoML-инфраструктуры увеличивает поверхность атак.

2) Оценка функций оптимизации может быть чрезвычайно дорогостоящей для сложных конвейеров машинного обучения и/или больших наборов данных.

3) Конфигурационное пространство часто бывает многомерным и структурно сложным (включающим сочетание непрерывных, категориальных и условных гиперпараметров). Кроме того, не всегда ясно, какие из гиперпараметров алгоритма необходимо оптимизировать и в каких диапазонах.

4) Обычно отсутствует доступ к градиенту функции потерь относительно гиперпараметров. Кроме того, другие свойства целевой функции (например, выпуклость и гладкость), часто используемые в классической оптимизации, в данном случае обычно неприменимы.

5) Нельзя напрямую оптимизировать качество обобщения, поскольку обучающие наборы данных имеют ограниченный размер.

6) Часто отсутствует ценная информация о влиянии различных гиперпараметров на конечное качество модели. Отсутствие объяснимости затрудняет доверие и понимание автоматизированного процесса HPO и его результатов. Кроме того, из-за иногда непрозрачной природы AutoML ошибки моделирования (внесенные сознательно или несознательно) очень трудно обнаружить.

Такой перечень проблем актуализирует необходимость формализации ряда взаимосвязанных задач для AutoML. А именно – задачи оптимизации гиперпараметров (задача AutoML), задачи совместной оптимизации выбора алгоритма (модели) обучения и гиперпараметров, (Combined Algorithm Selection and Hyperparameter - CASH), а также задачи *доверенного* автоматического машинного обучения (Trusted AutoML). Trusted AutoML включает в себя три подзадачи: «Transparency through Trust» – доверие через понимание принимаемых ML-моделью решений, «Trust through Robusticity» – доверие благодаря обеспечению надежности модели, «Trust through Security» – доверие посредством обеспечения информационной безопасности [13].

При формулировке задачи AutoML и CASH мы следуем изложения, формально представленному в [6]. И дополняем их в данной работе – делаем акцент на подзадаче Trusted AutoML в смысле Trust-through-Security, предлагая собственную формулировку этой подзадачи и общий протокол Trusted AutoML.

В практической части настоящей работы мы исследуем целесообразность применения предложенного протокола, решая задачу выбора оптимального ML-конвейера в защищенном (доверенном) и незащищенном режимах. Для этого, следуя поставленным формулировкам AutoML и Trusted AutoML, мы решаем задачу многомерной нелинейной регрессии для оценки реальной мощности независимой газовой турбины, используя открытый датасет Data-set for Independent Gas turbine for Electricity Generation [https://data.mendeley.com/datasets/6w3vy3ybhg/3].

Мы оцениваем затраченные вычислительные ресурсы (время и память) при обновлении конфигураций конвейеров ML-моделей регрессора, исходя из предположений, что среди 50 входных паромеров, имеющихся в представленном датасете и влияющих на реальную мощность газовой турбины, присутствуют такие, которые получаются в ходе функционирования компоненты цифрового двойника – рабочего колеса газовой турбины. В качестве промышленного объекта для цифрового двойника рассматривается газотурбинный двигатель (рис. 1), симуляция которого должна проходить в режиме «исследование-эксплуатация» в случаях обновления параметров его составляющих компонент (например, формы

рабочих лопаток турбины). На данном этапе влияние обновления указанных компонент не исследовалось.

Рис. 1. Газовая турбина SGT6-9000HL с увеличенным фрагментом ее лопаток (изображение с ресурса https://p3.aprimocdn.net/siemensenergy/f31eafe1-17e9-4ffd-961b-b0b300c968a1/GT-Portfolio-Brochure-2023-update_20231031_144ppi-pdf_Original%20file.pdf)

Для достижения данной цели авторами сформулированы в формализованном виде задача и общий протокол доверенного автоматического машинного обучения, проанализированы современные подходы к реализации конфиденциальных вычислений и распределенного машинного обучения, выявлены их особенности, достоинства и недостатки. Обоснован выбор TEE в пользу TEEs на базе виртуальных машинах (TEE on VM) для целевой задачи – автоматизированный ML-дизайн лопаток газотурбинных двигателей.

Приведены результаты моделирования конфиденциального распределенного обучения ML-регрессора. Показано, что использование методов автоматического распределенного обучения, позволяет добиться существенно большей эффективности обработки данных. В частности, показано, что примерно за равные временные интервалы обрабатывается кратно больше данных по сравнению с нераспределенными методами обучения, а также мы получаем более низкое значение функции потерь на валидационной выборке, что указывает на лучшее обобщение модели. Кроме того, реализация трансферного обучения на кластере Azure в доверенной виртуальной сети на базе TEEs существенно повышает надежность обработки данных с позиций конфиденциальности.

1. Формализации задачи доверенной автоматической оптимизации гиперпараметров и ML-алгоритмов

Традиционно проблему автоматического (без участия человека) машинного обучения AutoML рассматривают в рамках фиксированного вычислительного бюджета $B=\{b^{(i)}\}$. Составляющими бюджета B в реалистичных сценариях часто являются ограничения, такие как потребление памяти, время обучения, время прогнозирования, точность сжатой модели, энергопотребление и т.д. Как правило бюджет B включают в домен гиперпараметров и в ходе AutoML либо стараются его оптимизировать, либо не превысить. Формально проблему AutoML можно сформулировать следующим образом [6]:

Определение 1 (задача AutoML):

Пусть $D_{train}=\{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)\}$ обозначает обучающий набор данных, где $\mathbf{x}_i$ - вектор признаков, а y_i - соответствующее целевое значение, Соответственно, $D_{test}=\{(\mathbf{x}_{n+1}, y_{n+1}), \ldots, (\mathbf{x}_{n+m}, y_{n+m})\}$ – тестовый набор данных, взятый из того же базового распределения данных.

Пусть $Loss(\cdot,\cdot)$ –функция потерь, исследуемая в рамках бюджета B^*. Тогда задача AutoML заключается в том, чтобы (автоматически) получить точные прогнозы для набора тестов $\hat{y}_{n+1},\ldots,\hat{y}_{n+m}$:

$$\frac{1}{m}\sum_{j=1}^{m} Loss\left(\tilde{y}_{n+j}, y_{n+j}\right) \to \min_{\varepsilon\to 0,\ B\le B^*} \qquad (1)$$

Проблемы современного AutoML заключаются в следующем [6]:

1) Нет единого метода машинного обучения, который лучше всего работает со всеми наборами данных, а поисковое пространство огромно,

2) Некоторые методы машинного обучения (например, нелинейные машины опорных векторов - SVM) в решающей степени полагаются на оптимизацию своих гиперпараметров.

3) Если AutoML не контролировать, выбор подходящей модели может потребовать очень больших вычислительных ресурсов.

Проблема оптимизации гиперпараметров (Hyper Parameters Optimization -HPO), указанная второй в списке выше, на сегодня успешно решается с помощью байесовской оптимизации [8], которая в настоящее время является основным компонентом многих систем AutoM, таких как Microsoft NNI, Google Cloud AutoML H2O AutoML, AutoSKLearn, AutoKeras, DataRobot, VK Cloud ML Platform и др.

Первая проблема (выбора оптимального метода обучения) тесно переплетается с проблемой HPO, поскольку ранжирование алгоритмов обучения зависит от того, правильно ли настроены их гиперпараметры. К счастью, эти две проблемы могут быть эффективно решены как единая, структурированная, совместная задача оптимизации, получившая название CASH – Combined Algorithm Selection and Hyperparameter [6].

Определение 2 (задача CASH) Пусть $A = \{A^{(1)},..., A^{(R)}\}$ - набор алгоритмов, и пусть гиперпараметры каждого алгоритма $A^{(j)}$ имеют домен $\Lambda^{(j)}$. Пусть, по-прежнему, $D_{train}=\{(\mathbf{x}_1, y_1),...,(\mathbf{x}_n,y_n)\}$ - обучающий набор данных, который разбит на K групп перекрестной проверки (cross-validation) $\{D^{(1)}_{valid}, \ldots, D^{(K)}_{valid}\}$ и $\{D^{(1)}_{train}, \ldots, D^{(K)}_{train}\}$ такой, что $D^{(i)}_{train} = D_{train}\backslash D^{(i)}_{valid}$ для $i = 1, \ldots, K$.

Наконец, пусть $Loss(A^{(j)}_{\lambda}, D^{(i)}_{train}, D^{(i)}_{valid})$ обозначает функцию потерь, которую алгоритм $A^{(j)}$ достигает на $D^{(i)}_{valid}$ при обучении на $D^{(i)}_{train}$ с гиперпараметрами λ. Тогда алгоритм задачи комбинированного выбора алгоритма и оптимизации гиперпараметров (CASH) состоит в том, чтобы найти эффективную конфигурацию Q_j^* (совместный алгоритм и настройку гиперпараметров), которые минимизируют эти потери:

$$Q_j^* = \left\{A^*, \lambda^*\right\} \in \underset{A^{(j)}\in\mathbf{A},\ \lambda\in\Lambda^{(j)}}{\arg\min} \frac{1}{K}\sum_{i=1}^{k} Loss\left(A^{(j)}_{\lambda}, D^{(i)}_{train}, D^{(i)}_{valid}\right). \qquad (2)$$

Проблема CASH была впервые решена Торнтоном и др. [9] и использована системе Auto-WEKA, реализованной на языке Java (на текущий момент не поддерживается). При решении авторы использовали tree-based Bayesian optimization (TBO) [10], основная идея которого заключалась в том, что байесовская оптимизация при HPO использует вероятностную модель для определения взаимосвязи между настройками гиперпараметров и их измеряемой производительностью; затем TBO использует эту модель для выбора наиболее многообещающей настройки гиперпараметров (эксплуатирует вероятностную модель в известных хороших регионах, а

не исследует новые части пространства данных). Отметим, что байесовская оптимизация основана на модели гауссовых процессов (Gaussian process), которые лучше всего работают в задачах малой размерности с численными гиперпараметрами [6].

В свою очередь, модели на основе деревьев решений (например, Sequential model-based algorithm configuration - SMAC [11] работают на основе случайного леса (Random Forest – RF)) и более успешны в задачах большой размерности, структурированных и частично дискретных данных. Помимо использования RF, главной отличительной особенностью SMAC является то, что он позволяет проводить быструю перекрестную проверку (cross-validation), выполняя оценку один раз за обучающий цикл и заблаговременно отбрасывая неэффективные настройки гиперпараметров.

Третья, ранее упомянутая проблема AutoML заставляет по-новому взглянуть на сам подход к проведению исследований ML-моделей. Теперь в ходе исследования изначально важно выявить модели, которые непригодны для развертывания, иначе ценные вычислительные ресурсы будут потрачены впустую и могут привести к сбою конвейеров. Кроме того, распределенная инфраструктура AutoML с одной стороны повышает отказоустойчивость и надежность, но с другой – обостряет проблему информационной безопасности: большая вычислительная база существенно повышает поверхность атак (в том числе, состязательных [12]). Следовательно, требуется ряд априорных ограничений, которые должны контролировать AutoML не только с позиций производительности, но и информационной безопасности.

В таблице 1 представлены наиболее используемые в настоящее время инструменты AutoML (как проприетарные, так и open source). Некоторые из них автоматизируют как процесс HPO, так и поиск наиболее оптимальной архитектуры модели (Neural Architecture Search - NAS). Акцентируем внимание на том, что ни один инструмент не поддерживает конфиденциальные (доверительные) вычисления, способные реализовать режим криптозащиты «data-in-use», при котором поддерживается шифрование данных при их нахождении в оперативной памяти и/или в процессе вычислений. Это является потенциальной уязвимостью со стороны третьих лиц (Third Parties).

Очевидно, что нет необходимости в криптозащите «на лету» всей AutoML-инфраструктуры. Однако чувствительная информация (например, выявленные оптимальные конфигурации конвейеров ML-моделей) должна подвергаться такому типу защиты. Далее в данной работе мы формализуем и исследуем возможность эффективного использования TEE для устранения таких уязвимостей.

Для поиска Q_j^* компании затрачивают большое количество либо своих, либо арендованных системных и временных ресурсов, и считают их своим активом, подлежащим защите как от неправомерной модификации, так и прочтения. Как правило Q_j^* дополнительно подвергаются интерпретации, с целью оценки влияния различных гиперпараметров на конечную производительность модели, и верифицируются на предмет соответствия политикам информационной безопасности (ПолИБ).

Таблица 1

Современные инструменты AutoML

Имя	Открытость источника	Способ развертывания	Возможность режима криптозащиты			Функции AutoML	Примечание
			Data-at-Rest	Data in Transit	Data-in-Use		
Azure AutoML	Да	On-premise / Managed	Да	Да	Нет	HPO + NAS	Имеет веб-интерфейс
AutoGluon	Да	On-premise	Да	Да	Нет	NAS	Поддерживает штрафы за большой размер моделей
AutoKeras	Да	On-premise	Да	Нет	Нет	NAS,	В зависимости от сценария, имеет базовые показатели, которые исследуются в первую очередь
TPOT	Да	On-premise	Да	Нет	Нет	HPO	Формирует конвейеры ансамблей, поддерживает генетические алгоритмы
H2O Driverless.ai	Нет	On-premise	Да	Да	Нет	HPO.	Использует множество схем предварительной обработки и кодирования объектов
H2O.ai AutoML	Да	On-premise	Да	Да	Нет	HPO.	Имеет веб-интерфейс и встроенную оценку
Google Vision AutoML	Нет	Managed	Да	Да	Нет	NAS.	Поддерживает трансферное обучение + NAS, минималистичный пользовательский интерфейс и интегрированная оценка
DataRobot	Нет	On-premise / Managed	Да	Да	Нет	HPO.	Поддержка XAI, встроенная оценка, дружественный интерфейс.
AutoSKLearn	Да	On-premise	Да	Нет	Нет	HPO.	Открытый программный код

Это актуализирует задачу *доверенной* автоматической оптимизации гиперпараметров и ML-алгоритмов (Trusted AutoML), которая, как было отмечено ранее, включает в себя три подзадачи: «Transparency through Trust» – доверие через понимание принимаемых MLмоделью решений, «Trust through Robusticity» – доверие благодаря обеспечению надежности модели, «Trust through Security» – доверие посредством обеспечения информационной безопасности [13].

В данной работе акцент сделан на задаче Trusted AutoML в смысле Trust-through-Security, которую мы формулируем следующим образом:

Определение 3 (Trusted AutoML):

Пусть $\mathbf{P_t} = \{P_t^{(1)},..., P_t^{(L)}\}$ – политика информационной безопасности, сопровождающая функционирование цифрового двойника (блока цифрового двойника) на t-ом временном периоде его жизненного цикла, представленная иерархиями правил $P_t^{(l)}$, где t принимает дискретные значения, а $l = 1,..., L$. Обозначим $SB_t^{(l)}$ – минимальный вычислительный бюджет, необходимый для реализации $P_t^{(l)}$, а через $F(P^{(l)},SB^{(l)})$ – нормализованную функцию, отражающую уровень защищенности цифрового двойника (блока цифрового двойника).

Пусть $\mathbf{A} = \{A^{(1)},..., A^{(R)}\}$ - набор алгоритмов, и пусть гиперпараметры каждого алгоритма $A^{(j)}$ имеют домен $\Lambda^{(j)}$. Пусть, $D_{train} = \{(\mathbf{x}_1, y_1),...,(\mathbf{x}_n,y_n)\}$ - обучающий набор данных, который разбит на K групп перекрестной проверки (cross-validation) $\{D^{(1)}_{valid}, \ldots, D^{(K)}_{valid}\}$ и $\{D^{(1)}_{train}, \ldots, D^{(K)}_{train}\}$ такой, что $D^{(i)}_{train} = D_{train} \backslash D^{(i)}_{valid}$ для $i = 1, \ldots, K$.

Обозначим $Loss(A^{(j)}_{\lambda}, D^{(i)}_{train}, D^{(i)}_{valid})$ как функцию потерь, которую алгоритм $A^{(j)}$ достигает на $D^{(i)}_{valid}$ при обучении на $D^{(i)}_{train}$ с гиперпараметрами λ. Тогда задача Trusted AutoML состоит в том, чтобы найти эффективный ансамбль $\{F^*(\cdot,\cdot)|^{\infty}_{t=1}, Q_j^*\})$, соответствующий выражению (3):

$$\left\{F^*\left(P_t^{(l)}, SB_t^{(l)}\right), Q_j^*\right\}\Big|_{t=1}^{T} = \left\{A^*, \lambda^*, P_t^*, SB_t^*\right\} \in$$

$$\in \arg\left(\begin{array}{l} \min\limits_{A^{(j)}\in\mathbf{A},\ \lambda\in\Lambda^{(j)}} \dfrac{1}{K}\sum\limits_{i=1}^{k} Loss\left(A_{\lambda}^{(j)}, D_{train}^{(i)}, D_{valid}^{(i)}\right) + \\ +C\cdot\left(1-\max\limits_{l=\overline{1,L}} F\left(P_t^{(l)}, SB_t^{(l)}\right)\right)\end{array}\right), \quad (3)$$

где C – коэффициент регуляризации, T – горизонт планирования доверенной работы ML-модели, $SB^{(l)}$ – композитный аргумент, в состав которого входит размер доверенной вычислительной базы.

Интерпретируя вербально выражение (3), можно сказать, что задача Trusted AutoML заключается в поиске компромисса при выделении вычислительного бюджета между эффективностью процесса AutoML и эффективностью защиты инфраструктуры AutoML от несанкционированного доступа. Это сложная задача динамической оптимизации, с целью частичного решения которой мы предлагаем использовать доверительные вычислительные среды – TEE.

Поясним часть гиперпараметров, которые фигурируют в правилах политик безопасности $P_t^{(l)}$ в контексте распределенной машинного обучения j-го блока цифрового двойника при работе с доверительными вычислительными средами.

Среди таковых следует выделить ролевые права доступа для программных сущностей цифрового двойника (например, суррогатные модели вложенной (nested) оптимизации); сертифицированные типы алгоритмов шифрования и цифровой подписи, используемые на доверенной стороне: допустимые размеры криптографических ключей, способы доказательств гарантий безопасности и т.п.

Пусть федерально решается задача инкрементного машинного обучения (при котором ML-модель дообучается, работая в режимах «исследование-эксплуатации») в контексте оптимизации общей производительности цифрового двойника (блока цифрового двойника). Разделим эту задачу на две фазы: фаза обучения ML-модели и фаза эксплуатации ML-модели. Требуется контролировать безопасность вычислений на обеих этих фазах.

Фаза 1. Пусть вычислительному узлу i $\left(i = \overline{1,M}\right)$ требуется передать локальную ML-модель ϕ_i для (до)обучения на облачный сервер с более мощными вычислительными ресурсами, где $\Phi = \left\{\phi_i, \left\|D_{train}^{(i)}\right\|\right\}_{i=1}^{M}$ и $\Phi = \prod_{i=1}^{M}\left\{\phi_i, \left\|D_{train}^{(i)}\right\|\right\}$ – задача оптимизации общей производительности цифрового двойника (или его компонента). Чтобы обеспечить безопасность модели Φ и датасета D_{train} (или его подвыборки, передаваемой на дообучения модели), на сервере используются конфиденциальные (доверенные) вычисления таким образом, что перед передачей на сервер датасет (или передаваемый фрагмент его) $D_{train}^{(i)} \rightarrow D_{train}^{*(i)}$ шифруется, передается в TEE и там расшифровывается, а сервер в ходе обучения формирует зачищенную ML-модель $\Phi^* = \left(\left\{\phi_i^*, \left\|D_{train}^{*(i)}\right\|\right\}, \mathbf{\Gamma}\right)$, где ϕ_i^* – защищенная локальная ML-модель узла i, $\mathbf{\Gamma}$ – доказательство гарантии целостности модели и датасета (подвыборки датасета). При этом результатом обучения для узла i является $\mathbf{\Gamma}$ и $\tilde{\varphi}_i$ такая, что $\Phi^* \xrightarrow{k(i)} \tilde{\varphi}_i$, $k(i)$ – приватный ключ узла i.

Тогда протокол взаимодействия между узлом i и облачным сервером, оснащенным TEE, описывается следующей последовательностью действий.

1) Для узла i создается безопасный канал с анклавом путем удаленной аттестации (удаленная аттестация основана на асимметричной криптографии).

2) Узел i загружает ϕ_i в анклав.

3) Анклав отвечает кортежем $\left\{\phi_i^*, \mathbf{\Gamma}_0, ans\right\}$, где $\mathbf{\Gamma_0}$ подтверждает правильность загрузки ϕ_i и отвечает $ans=0$, если загрузка корректна, и $ans=1$ – в противном случае.

4) Если загрузка корректна, то узел i отправляет анклаву зашифрованный набор данных D^*.

5) Анклав, взаимодействуя с ненадежным облачным сервером. получает результат $\Phi^* = \left(\left\{\phi_i^*, \left\|D_{train}^{*(i)}\right\|\right\}, \mathbf{\Gamma}\right)$

6) Анклав возвращает узлу i результат $\tilde{\varphi}_i$

7) Узел i с помощью своего приватного ключа получает расшифрованную обученную локальную ML-модель $\Phi^* \xrightarrow{k(i)} \tilde{\varphi}_i$;

При необходимости обученная модель на сервере может начать эксплуатироваться.

Фаза 2. При эксплуатации обученной модели $\widetilde{\Phi} = \prod_{i=1}^{M} \left\{ \tilde{\phi}_i, \left\| D_{train}^{(i)} \right\| \right\}$, ML-модель хранится на облачном доверенном сервере в виде $\Phi^* = \left(\left\{ \tilde{\phi}_i^*, \left\| D_{train}^{*(i)} \right\| \right\}, \mathbf{\Gamma} \right)$ и позволяет узлу i с ограниченными ресурсами отправить запрос в форме сэмпла зашифрованных исходных данных $\mathbf{x}^*$ и получить прогноз $y^* = \widetilde{\Phi}^* (\mathbf{x}^*) = \prod_{i=1}^{M} \left\{ \widetilde{\phi^*}_i, \mathbf{x}^* \right\}$ и доказательство $\mathbf{\Gamma}$: $y^* = \left(\left\{ \phi_i^*, \mathbf{x}^* \right\}, \mathbf{\Gamma} \right)$. После чего узел i может получить окончательный открытый прогноз y: $y^* \xrightarrow{k(i)} y$.

2. Проблемы применения доверенных сред и платформ при AutoML цифровых двойников

Доверенные среды выполнения TEEs [7] – это надежная среда, обеспечивающая целостность данных, конфиденциальность данных и целостность кода. Они обычно реализуются на отдельных чипах или системах на кристалле (SoC – System-on-Chip). Основная их задача – предотвратить несанкционированный доступ или изменение приложений и данных во время вычислений, тем самым обеспечивая их защиту на протяжении всего жизненного цикла.

Основная архитектурная идея TEE построена на разделении вычислительного пространства на два мира «Normal World» и «Secure World» с контролируемым буфером ввода-вывода, который является частью интерфейса взаимодействия между мирами. Сервисы и приложения вне TEE не могут прочитать или изменить данные внутри TEE. Модель угроз конфиденциальных вычислений направлена на удаление или уменьшение неправомерных возможностей оператора-поставщика облачных услуг или других субъектов в домене клиента.

TEE также используются для защиты собственной бизнес-логики, аналитических функций, алгоритмов машинного обучения при распараллеливании моделей обучения или целых приложений. Они активно развиваются за рубежом и курируются группой GlobalPlatform, разработавшей спецификации GP Technology TEE System Architecture v1.3 (White Paper: May 2022 – GPD_SPE_009) и GlobalPlatform Technology Root of Trust Definitions and Requirements Version 1.1 (Public Release June 2018 Document Reference: GP_REQ_025).

Развитие TEEs идет по двум направлениям – построение TEE на процессах (таких как, SGX от Intel и на виртуальных машинах (например, SEV от AMD). [https://docs.switchboard.xyz/tee]. Сравнение основной функциональности TEE с модулями доверенных платформ (TPM) и обычных операционных систем (в терминологии доверенных вычислений, обозначаемых Rich или Real) представлено в таблице 2. Характерные особенности технологических подходов к построению TEE отражены в таблице 3.

Глубокий аналитический обзор особенностей функционирования зарубежных TEE сделан [14], из которого можно сделать следующие основные выводы.

Таблица 2

Сравнение основной функциональности доверительных сред и модулей

Характеристика	Rich (Real) OS	TMP	TEE на процессах	TEE на VMz
Устойчивость к взлому (HW Tamper Resistance)	X	**V**	**O**	**O**
Собственное изолированное выполнение приложений (Native Isolated App .Execution)	X	**X**	**V**	**V**
Отсутствие потребности в дополнительном оборудовании (No Additional Hardware)	V	**X**	**V**	**V**
Удаленная аттестация (Remove Attestation)	X	**V**	**V**	**X**
Встроенная система безопасного ввода/вывода (Native Secure I/O)	X	**X**	**X**	**V**
Встроенное безопасное хранилище (Native Secure Storage)	X	**O**	**V**	**V**
Аутентифицированная загрузка (Authenticated Boot)	X	**V**	**X**	**V**

Обозначения: X – не поддерживает; V – поддерживает; O – ограниченная поддержка

Таблица 3

Характерные особенности технологий построения TEE

Технология	**TEE на процессах** (Виртуальные машины с конфиденциальностью на уровне анклавов приложений)	**TEE на виртуальных машинах** (Конфиденциальность на уровне всей виртуальной машины)
Элемент управления	Выполняется управление кодом по строкам для рабочих нагрузок приложений разработчика	Работает на уровне виртуальной машины, что позволяет перенести существующие виртуальные образы, чтобы воспользоваться преимуществами операций конфиденциальной памяти
Модель безопасности	Доверенная среда создает анклавы Intel SGX, а приложения получают доступ к безопасным данным из анклава.	AMD SEV-SNP обеспечивает целостность и проверяемое шифрование памяти, используемой виртуальными машинами.
Память	До 256 GiB памяти на основе SKU.	До 672 GiB памяти на основе SKU
Трудозатраты	Требуется время для планирования и разработки настраиваемого решения для виртуальной машины	Достаточно переместить виртуализированные рабочие нагрузки в среду конфиденциальных вычислений.
Случаи использования	Настраиваемые приложения, созданные для конфиденциальных вычислений	Исторические и/или существующие приложения, требующие конфиденциальных вычислений без их донастройки

Современные популярные реализации TEE, такие как Intel SGX, Intel TDX, AMD SEV-SNP, ARM TrustZone, ARM Realms, IBM PEF пока не полностью соответствуют требованиям AutoML. С позиций AutoML-инфраструктуры, которая относится к высоконагруженным системам, TEE требуют существенных улучшений в силу того, что изначально применение TEE позиционировалось для мобильных и встраиваемых систем. Для хостовых и серверных систем исторически предпочтение отдавалось модулям доверенных платформ (Trusted Platform Module – TPM), имеющих скромные возможности к адаптивности контролируемых ими программных компонент в силу хрупкости регистров конфигурации платформы (PCR). Однако TPM имеют более надежные (hard/static) корни доверия. Потребность в перманентном обновлении и адаптации программного обеспечения и, в том числе, прогресс в области облачных технологий и искусственного интеллекта наметили тенденцию к синергии между TEE и TPM [6]. Исследования по разработке TEE для высоконагруженных систем, включая авторов данной работы, на текущий момент находятся в активной стадии.

В целом, способность TEE размещать и выполнять сторонние приложения и службы в своей защищенной среде является ключевой. Выбирая для экспериментов базовую линию развития TEE в пользу TEEonVMs для ее сопряжения с AutoML-инфраструктурой цифровых двойников, мы исходили из следующих положений:

TEE on VMs

Достоинства:

1) Двусторонняя изоляция между виртуальной машиной (Secure World) и обычной системой (Normal World).

2) Не налагает требований относительно специфики разработки программного обеспечения.

Недостатки:

1) Большой (в сравнении с TEE on process) размер вычислительной базы (TCB), включающий ОС, работающую внутри виртуальной машины.

2) Требует эмуляции ядра и оборудования внутри виртуальной машины и является относительно тяжеловесным.

TEE on process

Достоинства:

1) Меньший (в сравнении с TEE on VMs) размер доверенной вычислительной базы (Trusted Computing Base - TCB), поскольку доверенными являются только центральный процессор и компонент конкретного процесса

Недостатки:

1) Отсутствие двунаправленной изоляции

2) Необходимость разработки приложений специально для этого типа TEE

Как видно из указанных выше достоинств, TEE on Process имеют меньший (в сравнении с TEE on VMs) размер доверенной вычислительной базы, что в рассматриваемом контексте однозначно приветствуется. Однако, в ходе проведенного авторами данной работы дополнительного анализа TEE, выявлено следующее.

1) В модели угроз конфиденциальных и доверенных вычислений операционная система реального мира (RealOS) не пользуется доверием, поэтому делегирование управления ресурсами RealOS может привести к уязвимостям. Следовательно, требуется, чтобы анклав контролировал свои таблицы страниц и применял политики безопасной подкачки внутри анклава, а это приводит к увеличению TCB и усложнению анклава (для AutoML – существенному увеличению), что в результате увеличивает поверхность атак, делая ее сравнимой с TEE on VMs.

2) TEE on Process полагается на непрерывную физическую память для анклава. Следовательно, если несколько анклавов выполняются одновременно (что при распределенном AutoML происходит повсеместно), требование непрерывной физической памяти для каждого анклава может привести к фрагментации и потенциально чрезмерному использованию ресурсов.

3) Поскольку TEE on VMs включают гостевую ОС в TCB, они позволяют прозрачно запускать многопоточные приложения.

В результате, в настоящем исследовании для экспериментов были выбраны TEE on VMs.

3. Анализ способов уменьшения поверхности атаки для Trusted AutoML

Доверенная вычислительная база (TCB – Trusted Computing Base) [7] – это совокупность системных ресурсов (аппаратного и программного обеспечения), которая отвечает за поддержание *политики безопасности системы*. Компоненты внутри TCB считаются "критически важными". Если один компонент внутри TCB скомпрометирован, безопасность всей системы может быть поставлена под угрозу. Важным атрибутом TCB является то, что она способна защитить себя от компрометации любым оборудованием или программным обеспечением, которое не является частью TCB.

Размер TCB тесно связан с поверхностью атак на доверительные среды и платформы: чем меньше доверенная вычислительная база, тем выше безопасность. Это снижает риск воздействия разных уязвимостей, вредоносных программ, атак и злоумышленников. Следовательно, магистральным направлением уменьшения поверхности атак является минимизация размера TCB.

Кратко рассмотрим выявленные авторами способы уменьшить поверхность атак Trusted AutoML в существующих реализациях. Отметим, что их можно сгруппировать в три основные группы:

1) минимизация размера TCB за счет исключения из ее состава CPU (стремятся оставить только GPU);

2) использование на стороне GPU гомоморфного шифрования;

3) использование слепых вычислений (Blinded computing).

В частности, среди существующих реализаций TEEs прежде всего следует выделить Slalom [15], который использует гомоморфное шифрование, позволяющее безопасно выгружать основную часть вычислений для оценки ML-модели на ненадежный (с точки зрения доверенной среды) графический процессор. К сожалению, на сегодняшний день гомоморфное шифрование сопряжено с неприемлемыми накладными расходами при большом количестве вычислений, свойственных AutoML. Но направление, несомненно, перспективное и требует дополнительных исследований.

HIX [16] – реализация со специальным анклавом SGX, называемым GPU enclave без подключения к изоляции CPU. Авторы стремились снизить затраты на криптографические

преобразования за счет того, что шифрование и дешифрование происходят только при передаче данных на графический процессор и обратно. Однако, поскольку графический процессор не модифицирован архитектурно для разделения миров, невозможно выполнить удаленную проверку его рабочих нагрузок. Это означает, что злоумышленник, который может манипулировать каналом PCIExpress (PCIe) или заменить сам графический процессор, может нарушить безопасность TEE. Следовательно, TEE в целом не может выполнять значимую удаленную аттестацию, если оператору системы априори не доверяют.

В отличии от HIX реализация Graviton [17] добавляет функциональность TEE в GPU, поддерживающую удаленную аттестацию и шифрование данных при передаче. Теперь, поскольку GPU больше не совместим с обычными графическими процессорами, соответствующий недостаток в HIX устранен.

Слепое вычисление сочетает в себе программно-управляемый механизм с динамическим отслеживанием искажений исполняемого кода; Релевантных реализаций TEE на базе Blended computing для анализа авторам найти не удалось (в публикациях в основном предлагаются незавершенные проекты).

Мы в своих экспериментах выбрали более осторожную тактику, опираясь на TEE on VM и используя вместо GPU процессоры в конфигурации с гиперпоточностью, оцениваем конкретную прикладную задачу – исследуем настолько, решая задачу многомерной нелинейной регрессии с помощью *распределенного* Trusted AutoML можно повысить/понизить скорость и качество обучения при ограничениях на размер TCB. Структурная схема развернутого стенда исследований представлена на рисунке 2.

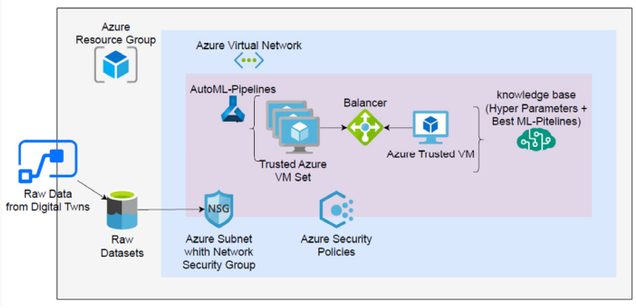

Рис. 2. Структурная схема используемого вычислительного стенда

Выполнялось в экспериментах распараллеливание данных [18], при котором датасет делится на подмножества в зависимости от количества узлов, доступных для обучения (одна и та же архитектура ML-модели используется во всех доступных узлах). Во время процесса обучения все узлы должны взаимодействовать друг с другом, чтобы обеспечить синхронизацию обучения на каждом узле (наиболее распространенная практика на текущий момент). Распараллеливание ML-моделей в рамках формируемых конвейеров для AutoML поддерживается по-умолчанию.

4. Результаты экспериментов

В качестве демонстрации моделирование проводилось без какого-либо рода чувствительной информации с использованием открытых датасетов, доступных на ресурсе Mendeley Data. Предварительные испытания для AutoML использовали датасет малого размера (Data-set for Independent Gas turbine for Electricity Generation. 3.01 Мб) [https://data.mendeley.com/datasets/6w3vy3ybhg/3]. Входными данными являются 50 параметров, представленных в ненормализованном (raw) виде (в датасете представлены почасовые измерения параметров за несколько суток), на выходе оценивалась реальная мощность (real power) [19] независимой газовой турбине (при заявленной мощности 5,68 МВт измеренные значения варьировались в разные периоды времени в диапазоне от 0 до 4,926 МВт). Интерпретация выявленных в ходе экспериментов релевантных признаков (features) из 50 исходных выходит за рамки данной статьи, поскольку акцент делался на контроль вычислительного бюджета.

Исследования включали оценку времени на формирование ML-конвейера регрессора четырьмя сериями экспериментов AutoML по 10 запусков в каждом в следующих режимах:

1) AutoML на одном узле вычислительного кластера (без дополнительного распараллеливания и без TEE);

2) AutoML с последовательным увеличением узлов вычислительного кластера от 2 до 4 (без TEE);

3) AutoML с последовательным увеличением узлов вычислительного кластера от 2 до 4 с выгрузкой в TEE всей вычислительной нагрузки машинного обучения;

4) AutoML с последовательным увеличением узлов вычислительного кластера от 2 до 4 с выгрузкой в TEE выявленных лучших ML-конфигураций;

Стэк технологий строился на базе Microsoft Azure —популярной платформе облачных вычислений, которая в качестве TEE on VM использует AMD SEV-SNP (GA) для шифрования всей памяти виртуальной машины; предлагает методы параллелизма как данных, так и моделей, а также поддерживает TensorFlow и Pytorch [https://neptune.ai/blog/distributed-training-frameworks-and-tools]. Пакет SDK Azure Python также позволяет взаимодействовать в любой среде Python, например, Jupyter Notebooks, Visual Studio Code и многих других.

В качестве формата представления ML-моделей выбран Open Neural Network Exchange (ONNX) —открытый стандартный формат для представления моделей машинного обучения, поддерживаемый большинством фреймворков (в том числе Pytorch, TensorFlow). Конфиденциальный сервер ввода/вывода ONNX [https://github.com/microsoft/onnx-server-openenclave], как компонент TEE, генерирует закрытый ключ (ECDH), который защищается внутри защищенного мира и используется для расшифровки входящих запросов на вывод. Клиент сначала получает открытый ключ сервера и отчет об аттестации, подтверждающий, что ключ был создан TEE. Затем он завершает обмен ключами для получения ключа шифрования для своего запроса.

Для формирования кластера Azure облачное решение реализует *доверенные виртуальные машины* Azure (Azure Trusted Virtual Machines), используя экземпляры серии Dv5.

Эти виртуальные машины предлагают до 96 виртуальных центральных процессов (vCPU) и 384 GiB оперативной памяти. Виртуальные машины серий Dv5 работают на процессорах Intel® Xeon® Platinum 8473C (Sapphire Rapids), Intel® Xeon® Platinum 8370C (Ice Lake) в конфигурации с гиперпоточностью, что обеспечивает лучшее соотношение цены и качества для большинства рабочих нагрузок. Данные процессоры оснащены турбо-тактовой частотой всех ядер 3,5 ГГц с технологией Intel® Turbo Boost, Intel® Advanced-Vector Extensions 512 (Intel® AVX-512) и Intel® Deep Learning Boost.

Ход эксперимента соответствует предложенному в разделе 1 общему протоколу Trusted AutoML. Следуя обозначениям в (3) A_i – элементы искомого ML-конвейера, ϕ_i – непосредственно есть ML-конвейер, в данном случае искомый ML-регрессор, $SB_t^{(1)}$ ограничение максимального размера памяти для виртуальной машины – 318 GiB, $SB_t^{(1)}$ – ограничение числа вычислительных кластеров – 4, горизонт к числу реализуемых правил политик безопасности отнесено требование использования в качестве доказательства гарантии безопасности – HMAC для ϕ_i. Горизонт планирования доверенной работы ML-модели T=1 сутки.

Чтобы обеспечить бесперебойную связь между экземплярами, все машины подключаются к одной и той же виртуальной сети, и создается *доверенная группа безопасности сети*, которая разрешает весь трафик от других узлов в кластере [20]. После обучения ML-модели гиперпараметры и ML-конфигурации сохраняются в доверенных виртуальных машинах.

Последовательность развертывания экспериментального стенда следующая (знаком (*) помечены шаги, выполненные только для 3 и 4 серии экспериментов):

1) Создание рабочей области ресурсов Azure

```
from azureml.core.workspace import Workspace
ws = Workspace.create(name='GZh_workspace',
    subscription_id='{subscription_id}',
    resource_group=' GZh_resourcegroup',
    create_resource_group=True,
    location='westeurope'
    )
```

2) Создание виртуальной сети для кластера виртуальных машин Azure

3) (*) Создание группы безопасности сети Azure

4) (*) Создание пары ключей Azure для SSH (закрытый/открытый)

5) Создание многоузлового вычислительного кластера машинного обучения Azure (начинаем с подготовки 1 узла и масштабируем до 4).

```
from azureml.core.compute import AmlCompute

aml_name = 'GZh_amlcompute'
try:
 aml_compute = AmlCompute(ws, aml_name)
print('Found existing AML compute context.')
except:
print('Creating new AML compute context.')
aml_config = AmlCompute.provisioning_configuration(vm_size = "Standard_D5_v2", min_nodes=1, max_nodes=4)
 aml_compute = AmlCompute.create(ws, name = aml_name, provisioning_configuration = aml_config)
    aml_compute.wait_for_completion(show_output = True)
```

6) (*) Создание виртуальных машин Trust Azure VM для распределенной точной настройки машинного обучения

7) (*) Настройка SSH

8) Использование AutoML для параллельного обучения пула моделей регрессии.

9) Оценка производительности лучшей модели

10) Выгрузка модели для перевода в нативный C-код (при необходимости) и/или развертывание модели в службах машинного обучения Azure

Результаты серии экспериментов отражены в таблице 4. В следствии асинхронного выполнения распределенных вычислений номера итераций в таблице 4 не следуют строгому порядку. Предельное время обучения для каждой итерации ограничено 20 мин. Количество итераций равно 25 по 10 запусков в каждой. При тестировании использовалась 10-кратная перекрестная проверка (cross-validation), в качестве функции потерь использовалось среднеквадратическое отклонение (RSME), а оценочной метрики r2_score - коэффициент детерминации.

В ходе AutoML было сформировано, обучено и оценено 25 конвейеров, из которых 9 лучших (с метриками R2_Score выше 0,9) представлены в таблице 5, а также на рисунках 3 и 4.

Как показывают эти результаты – распределенное обучение приводит к существенному сокращению времени обучения без потери (и даже к незначительному повышению) качества обучения независимо от использования (не использования) TEE.

В первой серии экспериментов за цикл обучения конвейер-лидер менялся трижды, соответственно дополнительные конфиденциальные вычисления в 4-серии экспериментов выполнялись только над четырьмя лучшими конфигурациями конвейеров, что в результате практически не снизило рабочие нагрузки по сравнению с серией 2 (без TEE), но гарантировало большее доверие в смысле Trust-through-Security. Дополнительные конфиденциальные вычисления над гиперпараметрами всех анализируемых конвейеров увеличили время вычислений в среднем на 1 минуту (или на 8,3%).

С целью проверки динамики ресурсозатрат на кластерах с 3 и 4 вычислительными узлами решено было оставить 4 лучших конвейера с сопоставимыми метриками R2_Score, а именно: MinMaxScaler + LightGBM (best R2_score); MinMaxScaler + GradientBoosting; RobustScaler+ LightGBM и StandardScalerWrapper+ LightGBM.

Результаты данных измерений, представленные на рисунке 5, позволяют сделать вывод, что, во-первых, распределенное машинное обучение приводит к снижению времени обучения пропорционально количеству вычислительных узлов. Во-вторых, с увеличением количества вычислительных узлов практически нивелируется разница по времени обучения при конфиденциальных вычислениях в TEE и без конфиденциальных вычислений.

Однако размер TCB продолжает оставаться значительным (равен объему памяти, занимаемой виртуальной машиной, в случае использованной AzureVM – до 318 GiB), динамическое снижение данного размера требует проведения дополнительных исследований.

Таблица 4

Результаты экспериментов распределенного AutoML

итера-ция	Конвейер (PIPELINE)	Серия 1 (1 вычислительный узел без TEE)			Серия 2 (2 вычислительных узла без TEE)			Серия 2 (2 вычислительных узла с TEE только Q-конфигурация.)			Серия 2 (2 вычислительных узла с TEE)		
		Время обучения	Метрика r2_score	Текущая луч-шая метрика	Время обучения	Метрика r2_score	Текущая луч-шая метрика	Время обучения	Метрика r2_score	Текущая луч-шая метрика	Время обучения	Метрика r2_score	Текущая луч-шая метрика
0	RobustScaler + ExtremeRandomTrees	0:10:46	0,9311	0,9311	0:07:11	0,9299	0,9299	0:07:31	0,9298	0,9298	0:08:28	0,9298	0,9298
10	StandardScalerWrapper+ LightGBM	0:12:46	0,9523	0,9523	0:08:32	0,9589	0,9589	0:08:39	0,959	0,959	0:09:40	0,9601	0,9601
11	StandardScalerWrapper+ RandomForest	0:12:20	0,9142	0,9523	0:08:12	0,9142	0,9589	0:08:11	0,9142	0,959	0:09:51	0,9142	0,9601
14	MinMaxScaler +RandomForest	0:12:36	0,9144	0,9523	0:08:33	0,9144	0,9589	0:08:34	0,9144	0,959	0:09:34	0,9141	0,9601
16	**MinMaxScaler + LightGBM**	**0:12:30**	**0,9643**	**0,9643**	**0:08:19**	**0,9666**	**0,9666**	**0:08:24**	**0,9661**	**0,9661**	**0:09:04**	**0,9662**	**0,9662**
7	RobustScaler+ LightGBM	0:17:41	0,9622	0,9643	0:11:41	0,9602	0,9666	0:11:41	0,9602	0,9661	0:12:01	0,9633	0,9662
20	MaxAbsScaler + RandomForest	0:12:06	0,9288	0,9643	0:08:01	0,9018	0,9666	0:08:00	0,9018	0,9661	0:08:50	0,911	0,9662
22	MinMaxScaler + GradientBoosting	0:11:38	0,9517	0,9643	0:08:00	0,9622	0,9666	0:08:04	0,9522	0,9661	**0:09:04**	**0,9622**	**0,9662**
17	MaxAbsScale+ LightGBM	0:17:20	0,9226	0,9643	0:12:10	0,9302	0,9666	0:12:11	0,9302	0,9661	0:13:11	0,9002	0,9662

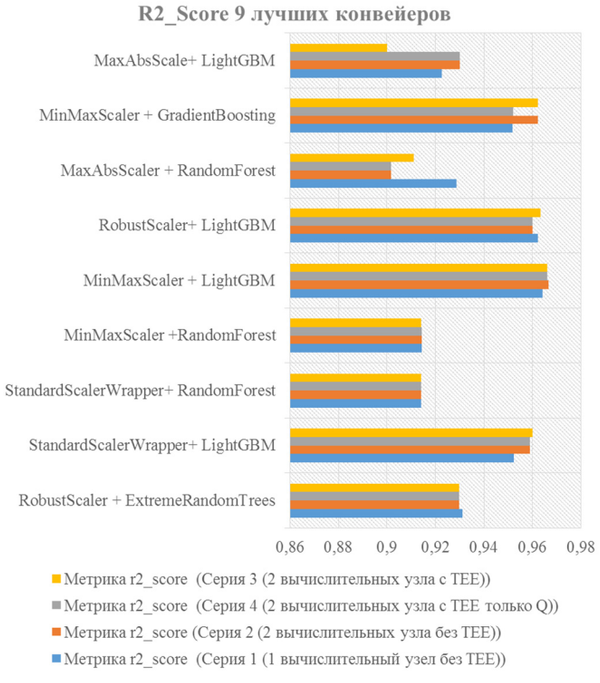

Рис. 3. Значения метрик R2_Score для 9 лучших обученных ML-конвейеров

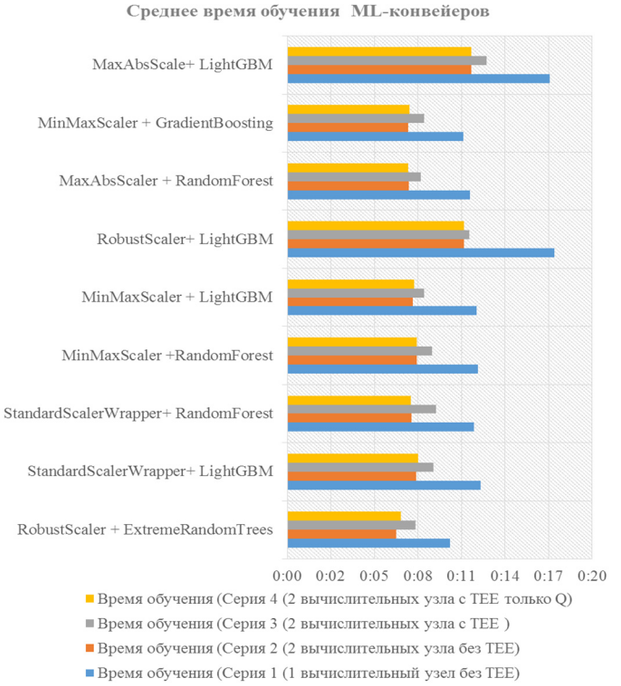

Рис. 4. Значения среднего времени обучения для 9 лучших обученных ML-конвейеров

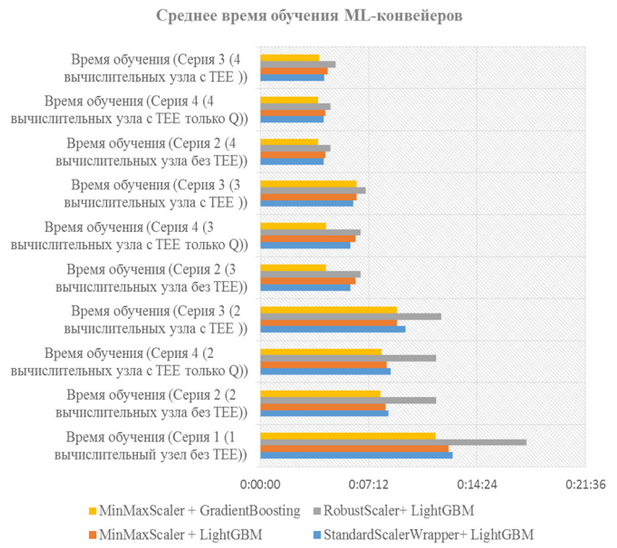

Рис. 5. среднее время распределенного обучения ML-конвейеров при масштабировании количества используемых вычислительных узлов

Заключение

В статье сформулированы задача и общий протокол *доверенного* автоматического машинного обучения, проанализированы современные подходы к реализации конфиденциальных вычислений и распределенного машинного обучения, Выявлено, что TEEs на виртуальных машинах для задачи автоматизированного машинного обучения является наиболее целесообразным выбором.

Приведены результаты моделирования конфиденциального распределенного обучения для задачи многомерной нелинейной регрессии, где в качестве прогноза модели оценивалась реальная мощность независимой газовой турбины. Показано, что использование методов автоматического распределенного обучения, позволяет добиться повышения эффективности обработки данных – время обработки снижается пропорционально числу вычислительных узлов при фиксированном размере TCB. Кроме того, реализация машинного обучения на кластере Azure в *доверенной* виртуальной сети на базе TEEs существенно повышает надежность обработки данных с позиций конфиденциальности.

Применение распределенного обучения на базе AutoML позволяет легко масштабировать обработку больших наборов данных. Эта масштабируемость имеет решающее значение для обработки реальных многомерных данных, генерируемых в процессе функционирования цифровых двойников.

Дальнейшие исследования авторов направлено на разработку механизмов автоматического масштабирования объемов доверенной вычислительной базы с целью снижения поверхности атак, а также разработку методик снижения TCB с применением гомоморфного шифрования.

Литература

1. *Patience N., Immerman D.* 2023 Global Trends in AI – WEKA. 2023. URL: https://www.weka.io/resources/analyst-report/2023-global-trends-in-ai/ (date access - 05.02.24)

2. *Kapteyn Michael G., Karen E. Willcox*. From Physics-Based Models to Predictive Digital Twins via Interpretable Machine Learning.” *ArXiv* abs/2004.11356. 2020: n. pag. URL: https://arxiv.org/abs/2004.11356 (дата обращения - 05.02.24)

3. *Chakraborty S., Adhikari S*. Machine learning based digital twin for dynamical systems with multiple time-scales // Computers & Structures. 2021. Т. 243. С. 106410, https://doi.org/10.1016/j.compstruc.2020.106410

4. *Morozov S.* Low-code platform to create, deploy and manage Digital Twins. 2022: URL: https://www.pseven.io/assets/files/publications/DUC2022/pSeven-Enterprise-Low-code-platform-to-create-deploy-and-manage-Digital-Twins.pdf.pdf (дата обращения - 05.02.24)

5. *Göppert A., Grahn L., Rachner J.* et al. Pipeline for ontology-based modeling and automated deployment of digital twins for planning and control of manufacturing systems // J Intell Manuf 34, 2133-2152. 2023. https://doi.org/10.1007/s10845-021-01860-6

6. *Hutter F. et al.* Automated Machine Learning, The Springer Series on Challenges in Machine Learning, https://doi.org/10.1007/978-3-030-05318-5_1

7. GP Technology TEE System Architecture v1.3 (White Paper: May 2022 – GPD_SPE_009) URL: https://globalplatform.org/wp-content/uploads/2022/05/GPD_SPE_009-GPD_TEE_SystemArchitecture_v1.3_PublicRelease_signed.pdf (дата обращения - 05.02.24)

8. *Bennett K.P., Kunapuli G., Jing Hu J.S.P.* Bilevel optimization and machine learning // Computational Intelligence: Research Frontiers, Lecture Notes in Computer Science, vol. 5050, pp. 25-47. Springer. 2008.

9. *Thornton C., Hutter F., Hoos H., Leyton-Brown K.* Auto-WEKA: combined selection and hyperparameter optimization of classification algorithms. In: Dhillon, I., Koren, Y., Ghani, R., Senator, T., Bradley, P., Parekh, R., He, J., Grossman, R., Uthurusamy, R. (eds.) The 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD’13), pp. 847-855. ACM Press. 2013.

10. *Mohr F., Wever M., Höllermeier E.* ML-Plan: Automated machine learning via hierarchical planning // Machine Learning, no. 107(8-10), pp. 1495-1515. 2018.

11. *Hutter F., Hoos H., Leyton-Brown K.* Sequential model-based optimization for general algorithm configuration // Proc. of LION-5, pp. 507-523. 2011.

12. *Фомичева С.Г., Беззатеев С.В.* Механизмы зашиты моделей машинного обучения от состязательных атак // T-Comm: Телекоммуникации и транспорт. 2023. Т. 17. № 10. С. 28-42. DOI: 10.36724/2072-8735-2023-17-10-28-42

13. *Fomicheva S., Bezzateev S.* Modification of the Berlekamp-Massey algorithm for explicable knowledge extraction by SIEM-agents // Journal of Physics: Conference Series. III International Conference on Metrological Support of Innovative Technologies (ICMSIT-III-2022). Krasnoyarsk, 2022. С. 52033.

14. *Muñoz Antonio, Rios Ruben, Roman Rodrigo, Lopez Javier*. 2023. A survey on the (in) security of Trusted Execution Environments. Computers & Security. 129. 103180. DOI: 10.1016/j.cose.2023.103180.

15. *Florian Tramer and Dan Boneh*. Slalom: Fast, Verifiable and Private Execution of Neural Networks in Trusted Hardware // Proceedings of the 2019 International Conference on Learning Representations. 2019. URL: https://openreview.net/forum?id=rJVorjCcKQ (date access - 05.02.24)

16. *Insu Jang* et al. Heterogeneous Isolated Execution for Commodity GPUs // Proceedings of the 2019 International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS). Providence, RI, USA: ACM, 2019, pp. 455-468. ISBN: 978-1-4503-6240-5. DOI: 10.1145/3297858.3304021

17. *Stavros Volos, Kapil Vaswani, Rodrigo Bruno*. Graviton: Trusted Execution Environments on GPUs // Proceedings of the 2018 USENIX Symposium on Operating Systems Design and Implementation (OSDI). Carlsbad, CA: USENIXAssociation, Oct. 2018, pp. 681-696. ISBN: 978-1-939133-08-3.

18. *Wang Hao* et al. Distributed Machine Learning with a Serverless Architecture // IEEE INFOCOM 2019 - IEEE Conference on Computer Communications. *2019: 1288-1296.*

19. *Zohuri Bahman*. 2015. Gas Turbine Working Principles. 10.1007/978-3-319-15560-9_7.

20. *Bezzateev S.V., Fomicheva S.G., Zhemelev G.A*. Techniques for Accelerating Algebraic Operations in Agent-based Information Security Systems // Wave Electronics and its Application in Information and Telecommunication Systems, WECONF – Conference Proceedings, 2023, 2023-May

TRUSTED AUTOMATIC MACHINE LEARNING IN THE OPERATION OF DIGITAL TWINS

Sergey V. Bezzateev, *University of Aerospace Instrumentations, Saint Petersburg, Russia,* ***bsv@aanet.ru***
Svetlana G. Fomicheva, *University of Aerospace Instrumentations, St. Petersburg, Russia,* ***levikha@mail.ru***
Georgiy A. Zhemelev, *Peter the Great St. Petersburg Polytechnic University, Saint Petersburg, Russia,* ***wws.dev@gmail.com***

Abstract
Purpose: The paper addresses the task of optimizing resource consumption in systems that integrate automatic machine learning methods with trusted computing environments in the context of updating configuration of ML-model pipelines required for functioning of a digital twin. Results: The task of trusted automatic machine learning has been formalized. The results of the analysis of modern approaches to the implementation of confidential computing and distributed machine learning are presented, their features, advantages and disadvantages are revealed. The results of modeling confidential distributed learning for elements of digital twins involved in the processes of adaptive gas turbine blades prototyping are presented. Practical relevance: It was shown that in the automatic solution of nonlinear multidimensional regression problems, the use of automatic distributed learning methods makes it possible to achieve a significant increase in data processing efficiency and ensure confidentiality of processing hyperparameters of ML-pipelines.

Keywords: *digital twins, automatic machine learning, trusted execution environments, optimization of computing resources.*

References

1. N. Patience, and D. Immerman, "2023 Global Trends in AI - WEKA". 2023. URL: https://www.weka.io/resources/analyst-report/2023-global-trends-in-ai/ (date access - 05.02.24).
2. Michael G. Kapteyn, and Karen E. Willcox, "From Physics-Based Models to Predictive Digital Twins via Interpretable Machine Learning," ArXiv abs/2004.11356 (2020): n. pag. URL: https://arxiv.org/abs/2004.11356 (date access - 05.02.24).
3. S. Chakraborty, S. Adhikari, "Machine learning based digital twin for dynamical systems with multiple time-scales," *Computers & Structures*. 2021. Vol. 243. P. 106410, https://doi.org/10.1016/j.compstruc.2020.106410 (date access - 05.02.24).
4. S. Morozov, "Low-code platform to create, deploy and manage Digital Twinsm" 2022: URL: https://www.pseven.io/assets/files/publications/DUC2022/pSeven-Enterprise-Low-code-platform-to-create-deploy-and-manage-Digital-Twins.pdf.pdf (date access - 05.02.24).
5. A. Goppert, L. Grahn, J. Rachner, et al., "Pipeline for ontology-based modeling and automated deployment of digital twins for planning and control of manufacturing systems," *J Intell Manuf*, no. 34, pp. 2133-2152. 2023. https://doi.org/10.1007/s10845-021-01860-6
6. F. Hutter et al., "Automated Machine Learning," The Springer Series on Challenges in Machine Learning, https://doi.org/10.1007/978-3-030-05318-5_1
7. GP Technology TEE System Architecture v1.3 (White Paper: May 2022 - GPD_SPE_009) URL: https://globalplatform.org/wp-content/uploads/2022/05/GPD_SPE_009-GPD_TEE_SystemArchitecture_v1.3_PublicRelease_signed.pdf (date access - 05.02.24)
8. K.P. Bennett, G. Kunapuli, Jing Hu, J.S.P., "Bilevel optimization and machine learning. In: Computational Intelligence: Research Frontiers," *Lecture Notes in Computer Science*, vol. 5050, pp. 25-47. Springer (2008).
9. C. Thornton, F. Hutter, H. Hoos, K. Leyton-Brown, "Auto-WEKA: combined selection and hyperparameter optimization of classification algorithms," In: Dhillon, I., Koren, Y., Ghani, R., Senator, T., Bradley, P., Parekh, R., He, J., Grossman, R., Uthurusamy, R. (eds.) The 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'13), pp. 847-855. ACM Press (2013).
10. F. Mohr, M. Wever, E. Hollermeier, "ML-Plan: Automated machine learning via hierarchical planning," *Machine Learning*, no. 107(8-10), pp. 1495-1515. 2018.
11. F. Hutter, H. Hoos, K. Leyton-Brown, "Sequential model-based optimization for general algorithm configuration," *Proc. of LION-5*, pp. 507-523. 2011.
12. S.G. Fomicheva, S.V. Bezzateev, "Defenses for machine learning models from adversarial attacks," *T-Comm*. 2023. Vol. 17. No. 10, pp. 28-42. DOI: 10.36724/2072-8735-2023-17-10-28-42
13. S. Fomicheva, S. Bezzateev, "Modification of the Berlekamp-Massey algorithm for explicable knowledge extraction by SIEM-agents," *Journal of Physics: Conference Series. III International Conference on Metrological Support of Innovative Technologies (ICMSIT-III-2022)*. Krasnoyarsk, 2022. P. 52033.
14. Munoz Antonio, Rios,Ruben, Roman Rodrigo, Lopez Javier, "A survey on the (in) security of Trusted Execution Environments. Computers & Security," 2023. 129. 103180. DOI: 10.1016/j.cose.2023.103180.
15. Florian Tramer and Dan Boneh, "Slalom: Fast, Verifiable and Private Execution of Neural Networks in Trusted Hardware," *Proceedings of the 2019 International Conference on Learning Representations*. 2019. URL: https://openreview.net/forum?id=rJVorjCcKQ (date access - 05.02.24).
16. Insu Jang et al., "Heterogeneous Isolated Execution for Commodity GPUs," *Proceedings of the 2019 International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS)*. Providence, RI, USA: ACM, 2019, pp. 455-468. ISBN: 978-1-4503-6240-5. DOI: 10.1145/3297858.3304021
17. Stavros Volos, Kapil Vaswani, and Rodrigo Bruno, "Graviton: Trusted Execution Environments on GPUs," *Proceedings of the 2018 USENIX Symposium on Operating Systems Design and Implementation (OSDI)*. Carlsbad, CA: USENIXAssociation, Oct. 2018, pp. 681-696. ISBN: 978-1-939133-08-3.
18. Wang Hao et al., "Distributed Machine Learning with a Serverless Architecture," *IEEE INFOCOM 2019 - IEEE Conference on Computer Communications*. 2019, pp. 1288-1296.
19. Zohuri, Bahman, "Gas Turbine Working Principles," 2015. 10.1007/978-3-319-15560-9_7.
20. S.V. Bezzateev, S.G. Fomicheva, G.A. Zhemelev, "Techniques for Accelerating Algebraic Operations in Agent-based Information Security Systems," Wave Electronics and its Application in Information and Telecommunication Systems, WECONF - Conference Proceedings, 2023.

Information about authors:
Sergey V. Bezzateev, *PhD, Docent, Head of the Department of Information Security, St. Petersburg University of Aerospace Instrumentations*
Svetlana G. Fomicheva, *PhD, Full Professor, Professor at the Department of Information Security, St. Petersburg University of Aerospace Instrumentations*
Georgiy A. Zhemelev, *Postgraduate student, Higher school of software engineering, Peter the Great St. Petersburg Polytechnic University*