УДК 004.89

ИНТЕГРАЦИЯ МЕХАНИЗМОВ ДОВЕРИЯ В СУЩЕСТВУЮЩИЕ ML-ФРЕЙМВОРКИ

Соколов И.В., студент гр. М3304, II курс
Университет ИТМО, г. Санкт-Петербург

Рост масштабов и сложности моделей машинного обучения, а также их внедрение в сферы, связанные с принятием социально значимых и потенциально необратимых решений – таких как здравоохранение, судебная экспертиза, финансовая оценка и автономные системы управления, – требуют не только высокой точности, но и институционального доверия к вычислительным процессам. При этом большинство распространённых ML-фреймворков, таких как TensorFlow, PyTorch, JAX и другие, изначально проектировались с приоритетом на производительность и гибкость, а не на верифицируемость или воспроизводимость. Поэтому возникает задача интеграции механизмов доверия непосредственно в существующие фреймворки машинного обучения, без существенного изменения их архитектуры и с сохранением совместимости с существующим экосистемным стеком. Настоящая работа посвящена анализу концептуальных и технических решений, направленных на поэтапное внедрение доверенных механизмов в стандартные ML-фреймворки [1].

Интеграция механизмов доверия в данном контексте означает включение в существующие фреймворки таких подсистем, которые позволяют формализованно контролировать поведение модели, её интерпретируемость, прозрачность данных, устойчивость к атакам и соответствие нормативно-этическим требованиям. Одним из базовых направлений является внедрение модулей для отслеживания происхождения данных (data lineage) и контроля их качества, включая автоматическое логгирование метаинформации, источников и способов трансформации выборки. Это повышает уровень доверия к модели как на этапе обучения, так и при эксплуатации, позволяя проводить аудит и воспроизводить поведение модели в ретроспективе.

Следующим уровнем интеграции выступают механизмы контроля корректности и устойчивости моделей. Сюда относятся средства автоматической оценки чувствительности модели к входным искажениям, анализ стабильности градиентов, модульные тесты устойчивости к дистрибутивным сдвигам. В рамках PyTorch и TensorFlow такие возможности могут быть реализованы как отдельные callback-модули в тренировочном

цикле или как плагины, подключаемые к системе логгирования (например, через TensorBoard или MLflow). Это позволяет расширить существующую экосистему без нарушения обратной совместимости и с возможностью гибкой настройки [2].

Особое значение имеет включение механизмов объяснимости. Современные фреймворки уже поддерживают интеграцию с XAI-инструментами, но в контексте доверия необходимо не просто наличие объяснения, а его верифицируемость и воспроизводимость. Это требует унификации форматов объяснений, возможности их сериализации и автоматической проверки соответствия предсказаний модели интерпретируемой логике. Интеграция таких механизмов может быть реализована через внутренние хуки в вычислительном графе, позволяющие перехватывать промежуточные представления модели и сопоставлять их с эталонными шаблонами или правилами [3].

С точки зрения инфраструктуры, необходимым элементом является поддержка форматов доверенных вычислений, включая интеграцию с системами безопасного выполнения (Trusted Execution Environments), библиотеками дифференциальной приватности (например, TensorFlow Privacy) и механизмами контроля версий моделей. Эти элементы должны быть встроены не только на уровне библиотеки, но и в пайплайны автоматизированного обучения (AutoML), CI/CD-системы и интерфейсы взаимодействия с API [4].

Наконец, важным направлением остаётся нормативная и этическая верификация. В существующие фреймворки могут быть встроены компоненты, автоматически проверяющие соответствие модели требованиям к недопущению дискриминации, прозрачности и устойчивости, в соответствии с региональными регламентами (GDPR, AI Act, ISO/IEC 24029 и др.). Это может быть реализовано как набор правил и политик, накладываемых на параметры модели и данные, с возможностью генерации аудиторских отчётов.

Таким образом, интеграция механизмов доверия в существующие ML-фреймворки требует системного подхода, охватывающего как вычислительный уровень, так и уровень взаимодействия с данными, интерфейсами и нормативной средой. Такой подход позволяет не создавать новые фреймворки с нуля, а эволюционно преобразовывать существующие инструменты в сторону доверенной инженерии ИИ. Это не только снижает барьер внедрения надёжных решений, но и способствует формированию общих стандартов доверия в машинном обучении как в научной, так и в промышленной среде [5].

Список литературы:

1. Bai, J. ONNX: Open Neural Network Exchange [Электронный ресурс] / J. Bai, F. Lu, K. Zhang [et al.]. – URL: https://onnx.ai/ (дата обращения: 09.06.2025).

2. Paszke, A. PyTorch: An Imperative Style, High-Performance Deep Learning Library / A. Paszke, S. Gross, F. Massa [et al.] // Advances in Neural Information Processing Systems. – 2019. – Vol. 32. – P. 8024–8035.

3. Abadi, M. TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems / M. Abadi, A. Agarwal, P. Barham [et al.] // arXiv preprint. – 2016. – arXiv:1603.04467.

4. Wolf, T. Transformers: State-of-the-Art Natural Language Processing / T. Wolf, L. Debut, V. Sanh [et al.] // Proceedings of the Conference on Empirical Methods in Natural Language Processing: System Demonstrations. – 2020. – P. 38–45. – DOI 10.18653/v1/2020.emnlp-demos.6.

5. OpenVINO Toolkit Documentation [Электронный ресурс]. – URL: https://docs.openvino.ai/ (дата обращения: 09.06.2025).