УДК 004.4
DOI



# ЭКСПЕРИМЕНТАЛЬНОЕ ИССЛЕДОВАНИЕ НАДЕЖНОСТИ ИНТЕЛЛЕКТУАЛЬНЫХ СИСТЕМ С ПРИМЕНЕНИЕМ МЕТОДОВ ОРГАНИЗАЦИИ ГЛОБАЛЬНО РАСПРЕДЕЛЕННОЙ АДАПТИВНОЙ ОБРАБОТКИ ПОТОКОВ ДАННЫХ

**Ермаков С. Р. ORCID ID 0000-0001-6082-1765**

*Федеральное государственное бюджетное образовательное учреждение высшего образования «Московский государственный технический университет радиотехники, электроники и автоматики – Российский технологический университет», Москва, Российская Федерация, e-mail: ermakov_s@mirea.ru*

Цель работы заключается в экспериментальном исследовании надежности интеллектуальных систем с применением методов организации глобально распределенной адаптивной обработки потоков данных путем оценки основных показателей обеспечения надежности информационных систем (SRE, Site Reliability Engineering) – задержки адаптивной обработки потоков данных, насыщения и степени разгрузки узлов. В исследовании проводится анализ поведения интеллектуальной системы, реализованной на основе трехуровневой архитектуры на экспериментальном стенде, при различных сценариях эксплуатации, включая резкие всплески пользовательской активности и потерю связи с центральным сервером. Проведено нагрузочное тестирование системы с моделированием сценариев резкого роста трафика и полной потери связи с облачным центром. Применены методы динамической разгрузки вычислительных задач (оффлоадинга) на клиентские устройства. В результате установлена нелинейная зависимость между насыщением туманного узла и интенсивностью выгрузки вычислений. Выявлено, что активация адаптивного оффлоадинга (с регулированием доли внешних вычислений от 30 до 78 %) позволяет стабилизировать время отклика системы на уровне 1,1–1,2 с при перегрузке, переводя ее в устойчивое квазистационарное состояние. Количественно подтвержден эффект мягкой деградации качества сервиса при разрыве соединения с облаком и показано, что снижение точности инференса происходит линейно и предсказуемо по мере устаревания данных, что обеспечивает сохранение уровня доступности системы.

**Ключевые слова: распределенные интеллектуальные системы, адаптивная обработка данных, надежность, туманные вычисления, федеративное обучение**

# EXPERIMENTAL STUDY OF INTELLIGENT SYSTEMS RELIABILITY USING GLOBALLY DISTRIBUTED ADAPTIVE DATA STREAM PROCESSING METHODS

**Ermakov S. R. ORCID ID 0000-0001-6082-1765**

*Federal State Budgetary Educational Institution of Higher Education "Moscow State Technical University of Radio Engineering, Electronics and Automation – Russian Technological University", Moscow, Russian Federation, e-mail: ermakov_s@mirea.ru*

The objective of this study is to experimentally investigate the reliability of intelligent systems using methods for organizing globally distributed adaptive data flow processing by assessing the key metrics of information system reliability assurance (SRE, Site Reliability Engineering) – adaptive data flow processing latency, node saturation, and node offloading. The study analyzes the behavior of an intelligent system, implemented using a three-tier architecture on an experimental setup, under various operational scenarios, including sudden surges in user activity and loss of connection to the central server. Load testing of the system was conducted, simulating scenarios of a sharp increase in traffic and complete loss of connection to the cloud center. Dynamic offloading of computational tasks to client devices was applied. As a result, a nonlinear relationship was established between fog node saturation and the intensity of computational offloading. It was found that activating adaptive offloading (with the proportion of external computations adjusted from 30 % to 78 %) stabilizes the system's response time at 1.1–1.2 seconds during overload, transitioning it to a stable, quasi-steady state. The effect of soft service degradation upon cloud connection loss was quantitatively confirmed, and it was shown that the decline in inference accuracy occurs linearly and predictably as data ages, ensuring the system's availability remains intact.

**Keywords: distributed intelligent systems, adaptive data processing, reliability, fog computing, federated learning**

## Введение

Стремительное развитие технологий искусственного интеллекта (ИИ) инициирует пересмотр традиционных подходов к проектированию архитектуры вычислительных систем. Классическая централизованная модель, предполагающая обработку всех запросов на мощностях удаленных дата-центров (серверов), в текущее время показывает все больше существенных ограничений при масштабировании [1]. Расширение дата-центров требует дорожающих компьютерных комплектующих, «силовое масштабирование» становится экономически и физически нецелесообразно для многих организаций, в том числе образовательных

[2]. Тем не менее конкуренция в области ИИ возрастает, все больше предприятий и вузов планируют внедрение собственных интеллектуальных систем и сервисов. Это в том числе связано с повышающимися сегодня требованиями к безопасности и предотвращению утечки данных. Технически ключевыми проблемами централизованных архитектур являются непредсказуемая сетевая задержка, высокая стоимость передачи больших объемов данных и критическая зависимость от доступности и скорости магистральных каналов связи, что отрицательно влияет на надежность вычислительных систем и может приводить к их неприменимости. Особенно остро данные проблемы проявляются в интеллектуальных системах, где требуется принятие решений в режиме реального времени [3].

В качестве перспективного решения автором рассматривается переход к организации глобально распределенной обработки потоков данных, основанной на концепции туманных вычислений и федеративного машинного обучения [4–6]. Суть данного подхода заключается в переносе части вычислительной нагрузки из центрального облака на промежуточные узлы, расположенные на границе сети, и непосредственно на клиентские устройства, что могло бы обеспечить масштабирование архитектуры путем оптимизации, а не расширения вычислительных характеристик оборудования.

**Цель исследования** – экспериментальное исследование надежности интеллектуальных систем с применением методов организации глобально распределенной адаптивной обработки потоков данных путем оценки основных показателей обеспечения надежности информационных систем (SRE, Site Reliability Engineering) – задержки адаптивной обработки потоков данных, насыщения и степени разгрузки узлов.

**Материалы и методы исследования**

Для проведения исследования была спроектирована и программно реализована архитектура распределенной интеллектуальной системы, структурно состоящая из трех функциональных уровней (рис. 1). Верхний уровень иерархии занимает *облачный сервер*, выполняющий роль глобального координатора и агрегатора. Его основной задачей является поддержание актуальной версии глобальной интеллектуальной модели, которая формируется путем объединения обновлений, поступающих от нижестоящих узлов. В модели облачный сервер реализует алгоритмы федеративного машинного обучения, принимая дифференциальные изменения весов моделей и рассчитывая метрику глобального качества.

Средний уровень образуют *туманные («вузовские») узлы*, которые располагаются в непосредственной близости к группам пользователей, например, в локальной сети организации или образовательного учреждения, но могут располагаться географически далеко друг от друга. Эти узлы берут на себя основную нагрузку по выполнению инференса – обработки запросов к интеллектуальной модели. Ключевой особенностью туманного узла в разработанной архитектуре является его способность к автономной работе. Узел хранит локальную копию модели и периодически, в асинхронном режиме, синхронизирует ее состояние с облачным сервером. Такой подход позволяет обслуживать запросы пользователей даже при временной недоступности глобальной сети [7, 8].

Туманные узлы включают в себя модель библиотек учебных заведений с коллекцией объемом 1000 документов (~6 ГБ размер файлов .pdf и .docx, размер чистого текста ~763 МБ, ~$1{,}6\times10^{8}$ токенов). Данный текстовый корпус моделирует реальный поток данных для последующей обработки и дообучения интеллектуальной большой языковой модели A-vibe [9], которая также установлена на туманных узлах.

Нижний уровень представлен *клиентскими (периферийными) устройствами*, под которыми понимаются персональные компьютеры или мобильные терминалы конечных пользователей. В предложенной архитектуре эти устройства не являются пассивными потребителями контента, а участвуют в вычислительном процессе. При высокой загрузке туманного узла часть задач по предварительной обработке данных, например токенизация или извлечение признаков из документов, передается на клиентские устройства. Это позволяет реализовать методы динамической разгрузки (оффлоадинга), который снижает нагрузку на компоненты туманного узла [10, 11].

Научная новизна исследования заключается в экспериментальном выявлении нелинейной зависимости между насыщением туманного узла и интенсивностью разгрузки вычислений, подтверждающей, что адаптивные методы динамической разгрузки могут переводить интеллектуальные системы в квазистационарное состояние со стабилизацией времени отклика при пиковых нагрузках; в определении границ эффективности гибридной обработки данных, обеспечивающих снижение нагрузки на вычислительные узлы за счет динамического перераспределения задач предварительной обработки в исследуемой архитектуре.

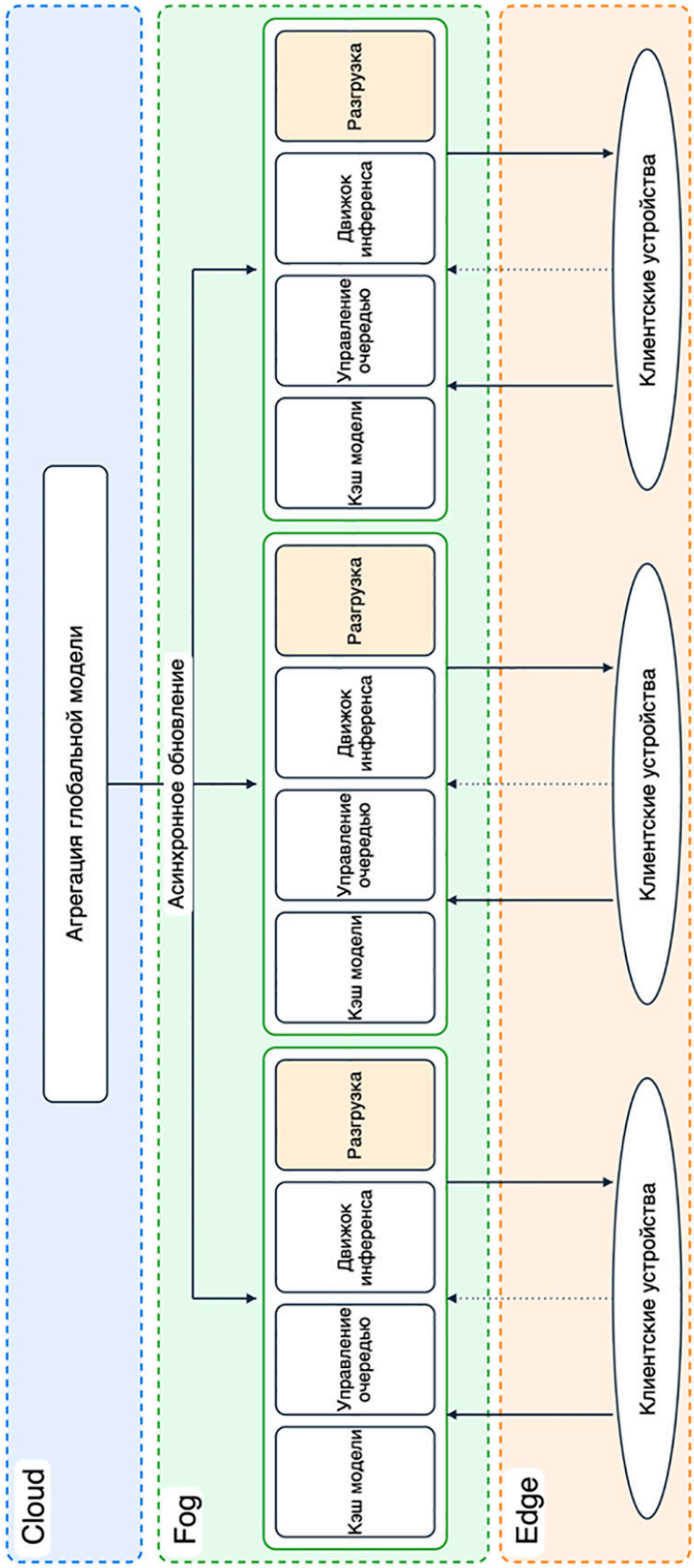


*Рис. 1. Предлагаемая архитектура интеллектуальных систем*
*Примечание: составлен автором по результатам данного исследования*

В общем виде архитектура и математическая модель системы основаны на концепции многокритериальной оптимизации качества обработки потоковых данных [12]. Для оперативного управления надежностью выполняется задача максимизации вероятности корректного ответа $P_{ok}$:

$$P_{ok} = \sigma\left(S_{net}\right) = \frac{1}{1+e^{-\left(aQ_{model} - bD_{task} - cT_{stale} - dP_{queue}\right)}}. \quad (*)$$

Здесь $Q_{model}$ отражает качество модели (аналог $\Phi_A$), $T_{stale}$ – показатель устаревания данных ($\Delta$), $P_{queue}$ характеризует нагрузку и задержки ($\Phi_P$), а $D_{task}$ – ресурсоемкость задачи ($\Phi_R$). Коэффициенты *a, b, c, d* задают чувствительность системы к различным видам возмущений и служат эмпирическими аналогами весов *w*.

Характеристики экспериментального стенда

| Параметр / Характеристика | Облачный сервер | Туманный узел | Периферийное устройство |
|---|---|---|---|
| Процессор (CPU/vCPU) | Intel Core i9-12900K (16 ядер / 24 потока) | Intel Core i7-11700 (8 ядер / 16 потоков) | Intel Core i5-10400 (6 ядер / 12 потоков) |
| Оперативная память (RAM) | 64 ГБ | 32 ГБ | 16 ГБ |
| Графическая подсистема (GPU) | NVIDIA GeForce RTX 3090 (24 ГБ vRAM) | NVIDIA GeForce RTX 3060 (12 ГБ vRAM) | Intel UHD Graphics 630 (Интегрированная) |
| Накопитель (SSD) | 1 ТБ | 512 ГБ | 256 ГБ |
| Операционная система (OS) | Ubuntu Server 24.04 LTS | Ubuntu Server 24.04 LTS / k3s | Ubuntu Desktop 24.04 LTS |

Примечание: составлена автором на основе технической конфигурации экспериментального стенда.

Экспериментальный стенд был развернут в среде оркестрации контейнеров Kubernetes (дистрибутив k3s), что позволило максимально приблизить условия тестирования к реальной эксплуатационной среде (таблица). Для генерации синтетической нагрузки использовался инструмент k6, позволяющий моделировать сложное поведение множества виртуальных пользователей. Система мониторинга была построена на базе стека Prometheus и Grafana, обеспечивая сбор метрик с дискретностью в одну секунду [13]. В ходе экспериментов фиксировались стандартные метрики производительности SRE.

**Результаты исследования и их обсуждение**

Механизм оффлоадинга реализован на Python с использованием NumPy (рис. 2); класс ReliabilityManager (метод calculate_p_ok) напрямую реализует модель (*), по телеметрии узла (очередь $P_{queue}$, рассинхронизация $T_{stale}$) вычисляет $P_{ok}$ для каждого запроса и при падении ниже порога надежности (threshold = 0.85) переключает обработку на оффлоадинг задач на клиентские устройства [14, 15].

В рамках нагрузочного тестирования были заданы несколько сценариев эксперимента: базовый сценарий для калибровки и получения эталонных показателей, линейный рост нагрузки для выявления предельной пропускной способности и точки деградации качества, а также стресс-сценарий всплеска нагрузки и отказ связи с облачным сегментом, представляющие основной интерес для анализа надежности.

В ходе эксперимента моделировался резкий восьмикратный рост входящей нагрузки (с 10 до 80 запросов в секунду), имитирующий начало массовой активности пользователей. На графике на рис. 3 зафиксирован момент нарастания пиковой нагрузки (ось Y) в интервале 10–35 тыс. итераций (ось X). Длина очереди ожидающих задач обработки продемонстрировала экспоненциальный рост, достигнув абсолютного максимума в 85 запросов в районе 37 тыс. итераций. Согласно (*) (параметр $P_{queue}$), это состояние соответствует критическому риску отказа в обслуживании.

Однако реализованная на экспериментальном стенде интеллектуальная система продемонстрировала корректную работу механизма адаптивной защиты.

```python
import numpy as np

class ReliabilityManager:
    def __init__(self, a=1.2, b=0.8, c=1.5, d=0.5):
        # Настроечные коэффициенты модели (эмпирические веса)
        self.a = a  # Вес качества модели (Q_model)
        self.b = b  # Вес сложности задачи (D_task)
        self.c = c  # Вес устаревания данных (T_stale)
        self.d = d  # Вес длины очереди (P_queue)
        self.threshold = 0.85  # Порог вероятности для оффлоадинга

    def sigmoid(self, x):
        return 1 / (1 + np.exp(-x))

    def calculate_p_ok(self, q_model, d_task, t_stale, p_queue):
        s_net = (self.a * q_model) - \
                (self.b * d_task) - \
                (self.c * t_stale) - \
                (self.d * p_queue)
        return self.sigmoid(s_net)

    def should_offload(self, task, node_state):
        q_model = node_state.current_accuracy      # Q_model
        d_task = task.complexity                   # D_task
        t_stale = node_state.time_since_last_sync  # T_stale
        p_queue = len(node_state.request_queue)    # P_queue
        p_ok = self.calculate_p_ok(q_model, d_task, t_stale, p_queue)
        if p_ok < self.threshold:
            return True  # Offload to Edge
        return False     # Process Locally
```

*Рис. 2. Программная реализация механизма динамической разгрузки*
*Примечание: составлен автором по результатам данного исследования*

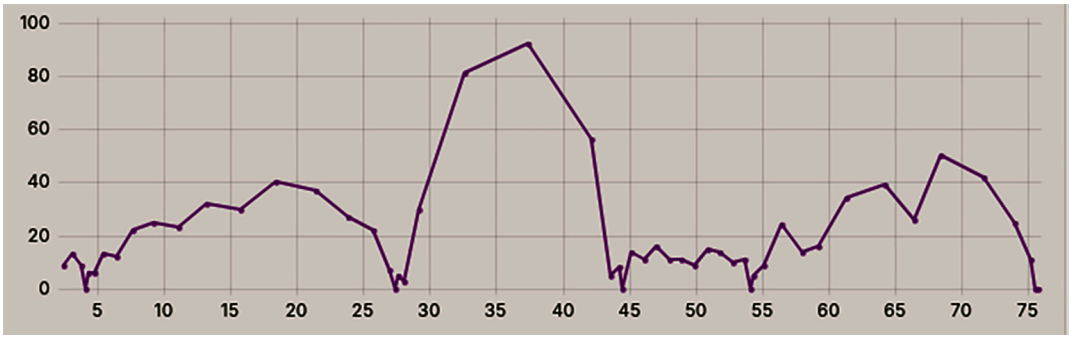


*Рис. 3. Насыщение узлов туманного уровня*
*Примечание: составлен автором по результатам данного исследования*

Сопоставление данных с графиком (рис. 4) показывает, что в момент роста очереди запросов выше порогового значения заранее произошла активация динамической разгрузки. Доля вычислительных задач предварительной обработки данных (токенизация, векторизация), передаваемых на периферийные устройства, резко возросла с фоновых 30–35 до 78 %.

Эффект от перераспределения нагрузки наблюдается мгновенно: начиная с 40-тысячной итерации длина очереди на туманном узле принудительно снижается и стабилизируется в диапазоне 5–20 запросов, несмотря на продолжающийся поток входящих запросов.

Анализ времени задержки узлов туманного уровня на рис. 5 подтверждает эффективность выбранной стратегии.

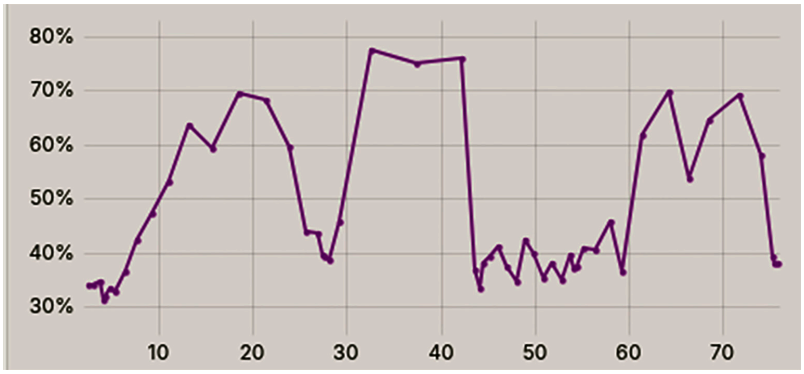


*Рис. 4. Степень разгрузки узлов туманного уровня*
*Примечание: составлен автором по результатам данного исследования*

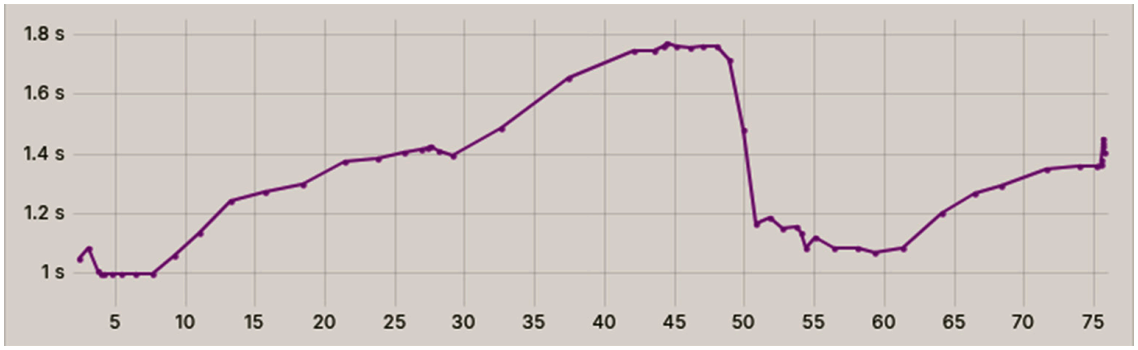


*Рис. 5. Средняя задержка узлов туманного уровня*
*Примечание: составлен автором по результатам данного исследования*

В фазе насыщения 95-й перцентиль задержки закономерно вырос с 1,0 до 1,78 с. Однако благодаря массовому сбросу задач на клиентские устройства система избежала неконтролируемого роста задержек. После стабилизации очереди задержка вернулась к значению 1,1–1,2 с, что является приемлемым показателем для интерактивных интеллектуальных систем.

Также в эксперименте исследовалась устойчивость системы к потере связи с облачным сервером с 60-тысячной итерации. При потере связи с облаком в традиционных системах сервис останавливается ($P_{ok} \rightarrow 0$), в предлагаемой же архитектуре $T_{stale}$ растет, вызывая мягкую деградацию по (*). Точность снижается плавно и постепенно, сервис работает автономно, сохраняя доступность.

## Заключение

В работе выполнено экспериментальное исследование надежности распределенных интеллектуальных систем с трехуровневой архитектурой и адаптивной разгрузкой вычислений на клиентские устройства при сценариях пиковых нагрузок и потери связи с облачным сегментом. Показано, что при восьмикратном росте интенсивности запросов (с 10 до 80 запросов/с) применение методов организации глобально распределенной адаптивной обработки потоков данных, увеличивающих долю внешних вычислений с 30–35 до 78 %, переводит систему в устойчивое квазистационарное состояние: очередь задач стабилизируется в диапазоне 5–20 запросов, а время отклика удерживается на уровне 1,1–1,2 с (после кратковременного роста p95-задержки до 1,78 с). Экспериментально подтвержден эффект «мягкой деградации» при разрыве связи с облаком: туманный узел сохраняет доступность сервиса за счет локальной копии модели, а качество инференса снижается плавно и предсказуемо по мере устаревания данных. Полученные результаты демонстрируют практическую применимость адаптивной разгрузки как механизма повышения надежности (устойчивости к перегрузке и сетевой изоляции) для интеллектуальных систем. В дальнейших исследованиях планируется автоматизировать настройку параметров математической модели для повышения точности и перено-

симости механизма управления в различных профилях нагрузки и инфраструктурных конфигурациях.

### Список литературы

1. Gundla N. K. Building Castles in the Cloud: Architecting Resilient and Scalable Infrastructure // International Journal of Computer Trends and Technology. 2024. Vol. 72. Is. 9. P. 77–92. URL: https://ijcttjournal.org/archives/ijctt-v72i9p113 (дата обращения: 04.02.2026). DOI: 10.14445/22312803/IJCTT-V72I9P113.

2. Strubell E., Ganesh A., McCallum A. Energy and Policy Considerations for Deep Learning in NLP // ACL Anthology. 2019. P. 3645–3650. URL: https://aclanthology.org/P19-1355/ (дата обращения: 04.02.2026). DOI: 10.18653/v1/P19-1355.

3. Villar-Rodriguez E., Arostegi Pérez M., Torre-Bastida A. I., Regueiro-Senderos C. Edge Intelligence Secure Frameworks: Current State and Future Challenges // Computers and Security. 2023. Vol. 130. Art. 103278. URL: https://www.researchgate.net/publication/370376605 (дата обращения: 04.02.2026). DOI: 10.1016/j.cose.2023.103278.

4. Al-Ansi A., Al-Ansi A. M., Muthanna A., Elgendy I. A., Koucheryavy A. Survey on Intelligence Edge Computing in 6G: Characteristics, Challenges, Potential Use Cases, and Market Drivers // Future Internet. 2021. Vol. 13. Is. 5. Art. 118. URL: https://www.mdpi.com/1999-5903/13/5/118 (дата обращения: 04.02.2026). DOI: 10.3390/fi13050118.

5. Heydari S., Mahmoud Q. H. Tiny Machine Learning and On-Device Inference: A Survey of Applications, Challenges, and Future Directions // Sensors. 2025. Vol. 25. Is. 10. P. 3191. URL: https://www.mdpi.com/1424-8220/25/10/3191 (дата обращения: 04.02.2026). DOI: 10.3390/s25103191.

6. Ермаков С. Р., Зыков С. В. Повышение эффективности потоковой обработки данных в интеллектуальной образовательной системе // Информационно-измерительные и управляющие системы. 2025. № 5. С. 41–53. URL: http://radiotec.ru/ru/journal/Information-measuring_and_Control_Systems/number/2025-5 (дата обращения: 04.02.2026).

7. Bandara E., Bouk S. H., Shetty S., Mukkamala R., Rahman A., Foytik P. SRE-Llama – Fine-Tuned Meta's Llama LLM, Federated Learning, Blockchain and NFT Enabled Site Reliability Engineering (SRE) Platform for Communication and Networking Software Services // 2025 7th International Conference on Blockchain Computing and Applications (BCCA). 2025. P. 344–351. URL: https://ieeexplore.ieee.org/document/11229660 (дата обращения: 04.02.2026). DOI: 10.1109/BCCA66705.2025.11229660.

8. Gogineni K., Suvizi A., Venkataramani G. A Survey on Large Language Model Acceleration based on KV Cache Management // Transactions on Machine Learning Research (TMLR). 2025. URL: https://huggingface.co/papers/2412.19442 (дата обращения: 04.02.2026).

9. AvitoTech. AvitoTech/avibe [Электронный ресурс] // Hugging Face. 2025. URL: https://huggingface.co/AvitoTech/avibe (дата обращения: 03.02.2026).

10. Palanisamy B., Xu J. Efficient and Resilient Edge Computing: Algorithms, Techniques and Research Opportunities // Proceedings of the 25th International Conference on Distributed Computing and Networking (ICDCN '24). 2024. P. 1–2. URL: https://dl.acm.org/doi/10.1145/3631461.3632515 (дата обращения: 04.02.2026). DOI: 10.1145/3631461.3632515.

11. Nemati A. M., Mansouri N. Resource allocation in fog computing: a survey on current state and research challenges // Knowledge and Information Systems. 2025. Vol. 67. P. 2091–2170. URL: https://link.springer.com/article/10.1007/s10115-024-02274-5 (дата обращения: 04.02.2026). DOI: 10.1007/s10115-024-02274-5.

12. Geldenhuys M. K., Scheinert D., Kao O., Thamsen L. Demeter: Resource-Efficient Distributed Stream Processing under Dynamic Loads with Multi-Configuration Optimization // Proceedings of the 15th ACM/SPEC International Conference on Performance Engineering (ICPE '24). 2024. P. 135–145. URL: https://dl.acm.org/doi/10.1145/3629526.3645048 (дата обращения: 04.02.2026). DOI: 10.1145/3629526.3645048.

13. Ермаков С. Р., Зыков С. В. Внедрение моделей машинного обучения в потоковой интеллектуальной образовательной системе // Современные наукоемкие технологии. 2025. № 2. С. 45–53. URL: https://top-technologies.ru/ru/issue/view?id=698 (дата обращения: 04.02.2026).

14. Zilic J., de Maio V., Ilager S., Brandic I. FRESCO: Fast and Reliable Edge Offloading With Reputation-Based Hybrid Smart Contracts // IEEE Transactions on Services Computing. 2025. Vol. 18. Is. 6. P. 3810–3823. DOI: 10.1109/TSC.2025.3520361.

15. Liang F., Liang C., Lu W., Zhao J., Zhang S. Decentralized and Network-Aware Task Offloading for Smart Transportation via Blockchain // Sensors. 2025. Vol. 25. Is. 17. Art. № 5555. URL: https://www.mdpi.com/1424-8220/25/17/5555 (дата обращения: 04.02.2026). DOI: 10.3390/s25175555.

**Конфликт интересов:** Автор заявляет об отсутствии конфликта интересов.

**Conflict of interest:** The author declares that there is no conflict of interest.