ISSN 2305-9052 (Print)
ISSN 2410-7034 (Online)

ВЕСТНИК ЮЖНО-УРАЛЬСКОГО ГОСУДАРСТВЕННОГО УНИВЕРСИТЕТА

BULLETIN OF THE SOUTH URAL STATE UNIVERSITY

СЕРИЯ

ВЫЧИСЛИТЕЛЬНАЯ МАТЕМАТИКА И ИНФОРМАТИКА

2025, том 14, № 4

SERIES

COMPUTATIONAL MATHEMATICS AND SOFTWARE ENGINEERING

2025, volume 14, no. 4

1943

ВЕСТНИК

ЮЖНО-УРАЛЬСКОГО ГОСУДАРСТВЕННОГО УНИВЕРСИТЕТА

2025
Т. 14, № 4

ISSN 2305-9052 (Print)
ISSN 2410-7034 (Online)

СЕРИЯ

«ВЫЧИСЛИТЕЛЬНАЯ МАТЕМАТИКА И ИНФОРМАТИКА»

Решением ВАК включен в Перечень научных изданий, в которых должны быть опубликованы результаты диссертаций на соискание ученых степеней кандидата и доктора наук

Учредитель — Федеральное государственное автономное образовательное учреждение высшего образования «Южно-Уральский государственный университет (национальный исследовательский университет)»

Тематика журнала:

- Вычислительная математика и численные методы
- Математическое программирование
- Распознавание образов
- Вычислительные методы линейной алгебры
- Решение обратных и некорректно поставленных задач
- Доказательные вычисления
- Численное решение дифференциальных и интегральных уравнений
- Исследование операций
- Теория игр
- Теория аппроксимации
- Информатика
- Искусственный интеллект и машинное обучение
- Системное программирование
- Перспективные многопроцессорные архитектуры
- Облачные вычисления
- Технология программирования
- Машинная графика
- Интернет-технологии
- Системы электронного обучения
- Технологии обработки баз данных и знаний
- Интеллектуальный анализ данных

BULLETIN
OF THE SOUTH URAL STATE UNIVERSITY
SERIES
"COMPUTATIONAL MATHEMATICS AND SOFTWARE ENGINEERING"

2025
Vol. 14, no. 4

ISSN 2305-9052 (Print)
ISSN 2410-7034 (Online)

Vestnik Yuzhno-Ural'skogo Gosudarstvennogo Universiteta.
Seriya "Vychislitel'naya Matematika i Informatika"

South Ural State University

The scope of the journal:

- Numerical analysis and methods
- Mathematical optimization
- Pattern recognition
- Numerical methods of linear algebra
- Reverse and ill-posed problems solution
- Computer-assisted proofs
- Numerical solutions of differential and integral equations
- Operations research
- Game theory
- Approximation theory
- Computer science
- Artificial intelligence and machine learning
- System software
- Advanced multiprocessor architectures
- Cloud computing
- Software engineering
- Computer graphics
- Internet technologies
- E-learning
- Database processing
- Data mining

Содержание

Contents

УДК 681.518.5, 004.89 **DOI: 10.14529/cmse250401**

КОНТРОЛЬ ДОСТОВЕРНОСТИ ПОКАЗАНИЙ СРЕДСТВ ИЗМЕРЕНИЙ ТЕХНИЧЕСКОГО МОНИТОРИНГА С ИСПОЛЬЗОВАНИЕМ КАСКАДА АВТОЭНКОДЕРОВ

© 2025 Д.В. Галышев, А.Д. Яковенко, О.Л. Ибряева, А.Л. Шестаков

Южно-Уральский государственный университет

(454080 Челябинск, пр. им. В.И. Ленина, д. 76)

E-mail: galyshevdv@susu.ru, iakovenkoad@susu.ru, ibriaevaol@susu.ru, a.l.shestakov@susu.ru

Поступила в редакцию: 17.08.2025

В статье предложен метод обеспечения достоверности измерений в системах технического мониторинга — каскадная модель C-LPC-AE, сочетающая извлечение информативных спектральных признаков на основе линейного предсказательного кодирования (LPC) и двухступенчатую архитектуру сверточных автоэнкодеров. Метод предназначен для диагностики состояния подшипников и одновременной верификации корректности работы измерительных датчиков, что особенно актуально в условиях цифровой индустрии, где требуется высокая автономность и надежность систем мониторинга. Первая ступень каскада, обученная на сигналах исправного подшипника при нормальном креплении датчика, выполняет обнаружение аномалий по ошибке реконструкции. Вторая ступень, обученная на данных с ослабленным креплением акселерометра, анализирует природу аномалии и позволяет дифференцировать аномалии подшипника от искажений сигнала, вызванных нарушением монтажа датчика. Ключевым преимуществом подхода является отсутствие необходимости в данных с реальными отказами оборудования: обучение осуществляется исключительно на легко воспроизводимых состояниях — нормальном режиме и моделируемой неисправности крепления. Эксперименты проведены на данных, полученных с испытательного стенда SpectraQuest, включающего подшипники с искусственно созданным дефектом внешнего кольца. Результаты демонстрируют высокую чувствительность модели к фактическим дефектам подшипника и нарушениям монтажа сенсора. Использование LPC-признаков обеспечивает компактное представление сигнала и снижает вычислительную нагрузку, что делает предложенный подход перспективным для внедрения в промышленные системы диагностики в реальном времени.

Ключевые слова: вибродиагностика, диагностика подшипников, ослабленное крепление датчика, автоэнкодер, обнаружение аномалий, технический мониторинг, линейное предсказательное кодирование, спектральный анализ, достоверность измерений.

ОБРАЗЕЦ ЦИТИРОВАНИЯ

Галышев Д.В., Яковенко А.Д., Ибряева О.Л., Шестаков А.Л. Контроль достоверности показаний средств измерений технического мониторинга с использованием каскада автоэнкодеров // Вестник ЮУрГУ. Серия: Вычислительная математика и информатика. 2025. Т. 14, № 4. С. 5–24. DOI: 10.14529/cmse250401.

Введение

С развитием цифровой индустрии и переходом к автономным системам технического мониторинга все большее значение приобретает достоверность данных, поступающих от средств измерений. Надежность принимаемых решений, особенно в условиях полной или частичной автоматизации, напрямую зависит от качества входных сигналов. Даже при использовании высокоточных алгоритмов диагностики и прогнозирования, недостоверные измерения могут привести к ложным срабатываниям, неверной интерпретации состояния оборудования и, как следствие, к экономическим потерям или аварийным ситуациям [1, 2].

Одной из типичных причин искажения данных является нарушение монтажа измерительных датчиков, в частности — ослабление крепления акселерометра. Такое нарушение

может изменить механическую жесткость узла, сместить резонансные частоты датчика и существенно повлиять на спектральный состав сигнала, даже при отсутствии дефектов в самом объекте диагностики. В результате система мониторинга может интерпретировать эти артефакты как признаки неисправности оборудования, что снижает ее доверие и эффективность.

Эта проблема особенно актуальна в задачах вибродиагностики, где подшипники качения, являясь критически важными элементами механических систем в авиационной, автомобильной и станкостроительной отраслях [3], требуют непрерывного мониторинга. Надежность их работы напрямую влияет на безопасность и экономическую эффективность эксплуатации оборудования [4]. Традиционные методы диагностики основаны на анализе вибрационных сигналов с применением подходов временной и спектральной обработки [5]. Методы временного анализа, как правило, опираются на статистические характеристики сигнала [6, 7], в то время как спектральный анализ позволяет выявлять частотные компоненты, характерные для конкретных типов дефектов [8–10]. Особое внимание уделяется анализу в области резонансной частоты датчика, где амплитуда сигнала, связанного с дефектом, значительно возрастает, что повышает чувствительность диагностики [11–13]. Однако традиционные подходы зачастую требуют ручного подбора признаков и остаются чувствительными к шумам и внешним помехам, что ограничивает их эффективность на ранних стадиях обнаружения неисправностей.

Современные подходы, основанные на машинном и глубоком обучении, демонстрируют высокую точность диагностики [14, 15], но сталкиваются с рядом ограничений: высокой вычислительной сложностью [16, 17], необходимостью в больших объемах размеченных данных и чувствительностью к шумам и артефактам [18]. Методы аугментации данных, переноса обучения и генеративных моделей [19–22] частично решают проблему нехватки обучающих выборок, но по-прежнему требуют доступа к данным с реальными дефектами.

В этом контексте особый интерес представляют методы обнаружения аномалий [23–25], в частности — автоэнкодеры, обучающиеся только на нормальных режимах работы [26–28]. Однако и они подвержены ложным срабатываниям, возникающим из-за шума, ограниченного объема обучающих данных или нестандартных режимов работы [29]. Одним из факторов, провоцирующих ложные срабатывания, являются искажения, вызванные неисправностью самого измерительного оборудования. Так, например, нарушение крепления акселерометра может изменить его резонансную частоту [1, 2], что приводит к искажению спектрального состава сигнала и, как следствие, снижает повторяемость и надежность исходных данных, на которых впоследствии строится весь процесс обработки.

В данной работе предлагается каскадная модель C-LPC-AE, направленная на обеспечение достоверности измерений в системах технического мониторинга. Подход сочетает линейное предсказательное кодирование для компактного представления спектральных данных и двухступенчатую архитектуру автоэнкодеров, где первая модель выявляет отклонения от нормы, а вторая верифицирует корректность работы измерительного датчика. Обучение осуществляется без привлечения данных о реальных отказах оборудования, что делает метод практичным для промышленного применения. Для предварительной обработки применяется метод линейного предсказания (LPC), позволяющий сжать спектральную информацию до компактного набора коэффициентов.

Предложенный метод решает три ключевые задачи: обеспечение достоверности измерений, снижение вычислительной сложности за счет компактных LPC-признаков и выявление

артефактов, возникающих вследствие сбоев в измерительном оборудовании. Результаты могут быть востребованы не только в вибродиагностике, но и в более широком классе систем мониторинга, где критически важно различать реальные неисправности оборудования и искажения, вызванные нарушениями в работе измерительной аппаратуры.

Основные результаты и вклад данной работы заключаются в следующем:

1. Разработана модель обработки вибросигналов, объединяющая LPC-анализ и каскадную архитектуру нейросетей, что позволяет разделять аномалии в состоянии оборудования и в работе средств измерений.
2. Предложен двухэтапный алгоритм верификации данных, не требующий примеров реальных отказов оборудования для обучения.
3. Предложен метод, сочетающий спектральный анализ и линейное предсказательное кодирование (LPC), обеспечивающий компактное и информативное представление данных при низкой вычислительной нагрузке.
4. Экспериментально подтверждена эффективность подхода в дифференциации двух типов отклонений: дефектов подшипника и искажений сигнала, вызванных нарушением монтажа акселерометра, что критически важно в автоматических системах контроля.
5. Предложенный метод соответствует требованиям цифровой индустрии к автономности, надежности и вычислительной эффективности, что делает его перспективным для внедрения в интеллектуальные средства измерений нового поколения.

Статья организована следующим образом. В разделе 1 описывается предложенный метод C-LPC-AE, включающий этапы предобработки сигналов на основе линейного предсказательного кодирования, извлечения компактных признаков и построения каскадной архитектуры автоэнкодеров для дифференциации аномалий. В разделе 2 представлены экспериментальные данные, описывается установка SpectraQuest, процедура формирования выборок и этапы выделения LPC-признаков, включая выбор оптимального порядка модели и визуализацию признакового пространства. В разделе 3 приведены параметры архитектуры автоэнкодеров и результаты тестирования каскадной модели на различных состояниях подшипника и датчика. В заключении обобщаются полученные результаты и обсуждается применимость предложенного подхода для автоматического контроля достоверности показаний средств измерений в условиях цифровой индустрии.

1. Предлагаемый метод каскада автоэнкодеров с признаками линейного предсказательного кодирования C-LPC-AE

1.1. Линейное предсказательное кодирование

Для извлечения информативных признаков из временного сигнала в предлагаемом методе применяется линейное предсказательное кодирование (LPC, Linear Predictive Coding). Этот подход широко используется при обработке акустических и вибрационных сигналов и зарекомендовал себя, в частности, в задаче классификации дефектов подшипников, где позволил сжать исходный амплитудный спектр до 50 коэффициентов, тем самым значительно снизив вычислительную сложность модели [30].

Ключевым предположением метода LPC является предположение о том, что n-й образец временного ряда можно аппроксимировать взвешенной суммой Q предыдущих образ-

цов [31]:

$$\hat{s}[n] = -\sum_{i=1}^{Q} a_i * s[n-i], \tag{1}$$

где s — временной ряд, Q — порядок модели, а a_i — коэффициенты модели. Таким образом, предсказание в любой момент времени представляет собой линейную комбинацию Q предыдущих отсчетов. Остаточный сигнал определяется как:

$$e[n] = s[n] - \hat{s}[n]. \tag{2}$$

Задача сводится к нахождению коэффициентов a_i, минимизирующих энергию остаточного сигнала:

$$p = \sum e^2[n] = \sum_n \left(s[n] + \sum_{i=1}^{Q} a_i * s[n-i] \right)^2. \tag{3}$$

Для вычисления оптимальных коэффициентов частные производные функции p по a_i приравниваются к нулю:

$$\frac{\partial p}{\partial a_k} = 2 \sum_n \left(s[n] + \sum_{i=1}^{Q} a_i * s[n-i] \right) s[n-k], \tag{4}$$

где $k = \overline{1, Q}$.

Уравнение (4) можно переписать в виде системы Q линейных уравнений Юла—Уокера, выраженных через автокорреляционные функции:

$$\sum_n s[n]s[n-k] + \sum_{i=1}^{Q} a_i \sum_n s[n-i]s[n-k] = 0 \tag{5}$$

или, эквивалентно,

$$\sum_{i=1}^{Q} a_i R[i-k] = -R[k] \tag{6}$$

для $k = \overline{1, Q}$, где $R[m] = \sum_n s[n]s[n-m]$ — автокорреляция сигнала с лагом m.

В матричной форме система записывается как:

$$Ra = -r, \tag{7}$$

где

$$\mathbf{R} = \begin{pmatrix} R[0] & R[1] & \cdots & R[Q-1] \\ R[1] & R[0] & \cdots & R[Q-2] \\ \vdots & \vdots & \ddots & \vdots \\ R[Q-1] & R[Q-2] & \cdots & R[0] \end{pmatrix}, \tag{8}$$

$$\mathbf{a} = \begin{bmatrix} a_1 & a_2 & \cdots & a_Q \end{bmatrix}^T, \tag{9}$$

$$\mathbf{r} = \begin{bmatrix} R[1] & R[2] & \cdots & R[Q] \end{bmatrix}^T. \tag{10}$$

Матрица $\mathbf{R}$ имеет структуру Теплица, что позволяет эффективно решать систему (7) с помощью рекурсии Левинсона—Дарбина. Этот метод последовательно строит модели от

первого до Q-го порядка, обновляя коэффициенты на каждом шаге с вычислительной сложностью $\mathcal{O}(Q^2)$ вместо $\mathcal{O}(Q^3)$ при прямом обращении матрицы.

1.2. Предобработка сигналов

Сигналы акселерометра по каждой из трех осей предварительно преобразуются в амплитудный спектр с использованием быстрого преобразования Фурье. Для дальнейшего анализа выбирается ограниченный диапазон частот: нижняя граница подбирается так, чтобы исключить низкочастотные составляющие, как правило, обусловленные механическими шумами и вибрациями, не связанными с состоянием подшипника; верхняя граница определяется с запасом, чтобы гарантировать покрытие резонансной частоты датчика.

Полученный спектр содержит значительное число коэффициентов, что затрудняет их прямое использование в моделях машинного обучения из-за высокой размерности и избыточности. Для снижения вычислительной сложности и извлечения информативных признаков спектральные данные далее обрабатываются методом линейного предсказательного кодирования (LPC), описанным в разделе 1.1. Этот шаг позволяет сжать спектральную информацию до фиксированного набора LPC-коэффициентов, переходя от обработки длинных спектров к более компактному и устойчивому к артефактам представлению. Полученные таким образом коэффициенты для каждой из осей датчика объединяются в единую матрицу:

$$\mathbf{S} = \begin{pmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,p} \\ a_{2,1} & a_{2,2} & \cdots & a_{2,p} \\ a_{3,1} & a_{3,2} & \cdots & a_{3,p} \end{pmatrix}, \tag{11}$$

где p — порядок модели LPC.

В зависимости от конфигурации измерительного узла для анализа может использоваться как одна, так и несколько осей акселерометра. В данной работе рассматривается трехосевой вариант как наиболее общий: использование всех трех каналов позволяет задействовать избыточность данных, повысить устойчивость к шуму и улучшить способность модели выявлять аномальные состояния. При необходимости методика может быть адаптирована и для одноосевого сенсора.

1.3. Каскад автоэнкодеров

После преобразования сигналов акселерометра и вычисления LPC-коэффициентов объединенная матрица признаков подается на вход первого автоэнкодера. Данная модель обучается исключительно на данных нормальных рабочих режимов оборудования и предназначена для первичного обнаружения аномалий. Для определения факта аномалии применяется порог, рассчитываемый по правилу пяти сигм:

$$\text{Threshold} = \mu_{train} + 5\sigma_{train}, \tag{12}$$

где μ_{train} — среднее значение ошибки восстановления на обучающей выборке, а σ_{train} — ее стандартное отклонение. Если ошибка восстановления превышает заданный порог, считается, что в сигнале зафиксирована аномалия, и данные передаются на обработку второму автоэнкодеру.

Второй автоэнкодер выполняет функцию оценки достоверности показаний средства измерения и обучается на сигналах с ослабленным креплением датчика, что позволяет отличать реальные дефекты подшипника от искажений, вызванных некорректной установ-

кой акселерометра. Таким образом, осуществляется верификация корректности измерений и исключение ложных диагностических заключений, вызванных сбоями в работе измерительной системы, что особенно важно в системах технического мониторинга в условиях цифровой индустрии, где критичны надежность и автоматический контроль достоверности поступающих данных.

Если ошибка восстановления на втором автоэнкодере превышает порог (12), ситуация интерпретируется как наличие реального дефекта подшипника. В противном случае аномалия, зафиксированная первым автоэнкодером, рассматривается как следствие неправильного крепления датчика.

Структура предлагаемой модели каскада автоэнкодеров показана на рис. 1. Ее ключевое преимущество заключается в том, что обучение обеих моделей (на нормальном состоянии и на состоянии с ослабленным креплением) возможно на данных, полученных в реальных условиях эксплуатации, без необходимости собирать труднодоступные и редко встречающиеся записи о фактических отказах.

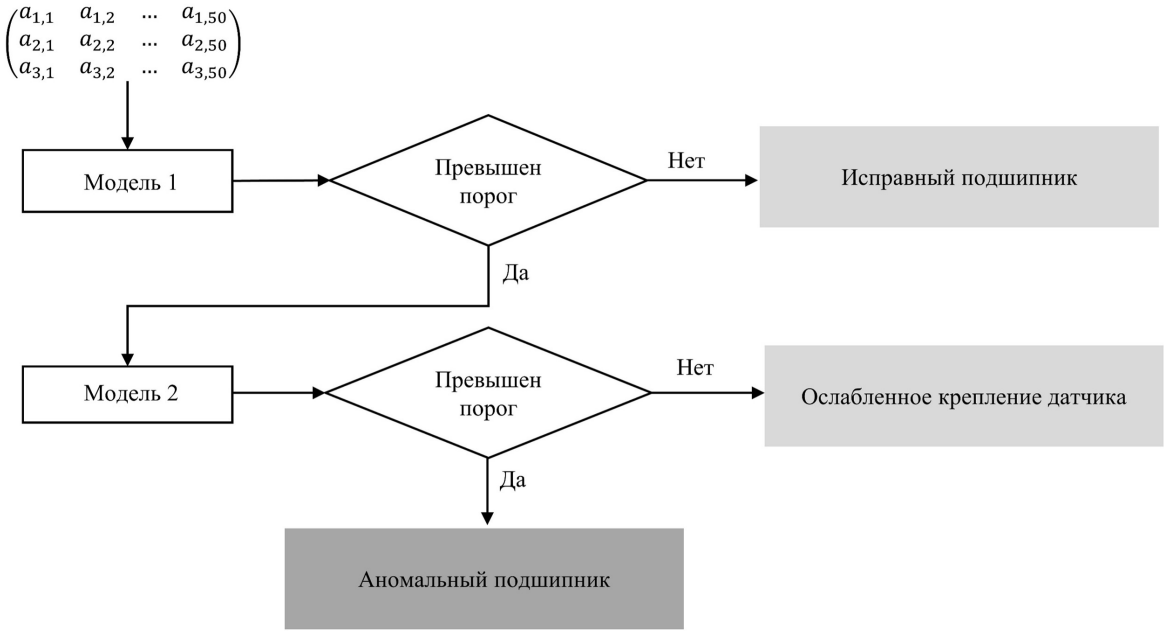

Рис. 1. Схема предлагаемой модели каскада автоэнкодеров

Следует учитывать, что ошибка восстановления каждого из автоэнкодеров может содержать случайные колебания даже при отсутствии реальных отклонений в сигнале. Это может привести к ложным срабатываниям системы. Для повышения надежности диагностики и снижения влияния таких колебаний производится сглаживание ошибки реконструкции с использованием скользящего окна.

2. Постановка эксперимента и выделение LPC признаков из данных

2.1. Экспериментальная установка

Экспериментальные данные для исследования были получены на испытательной установке SpectraQuest, оснащенной трехосевым вибропреобразователем AP2038P-100 с часто-

той установочного резонанса в осевом направлении 35 кГц. В рамках эксперимента использовались два дюймовых подшипника: один исправный, который служил эталоном, и один с искусственно введенным дефектом внешнего кольца, что позволяло моделировать аномальное состояние подшипника. Частота вращения подшипников в ходе испытаний фиксировалась на уровне 50 Гц, что соответствует номинальному режиму работы двигателя. Опора, на которой установлен диагностируемый подшипник и акселерометр, показана на рис. 2.

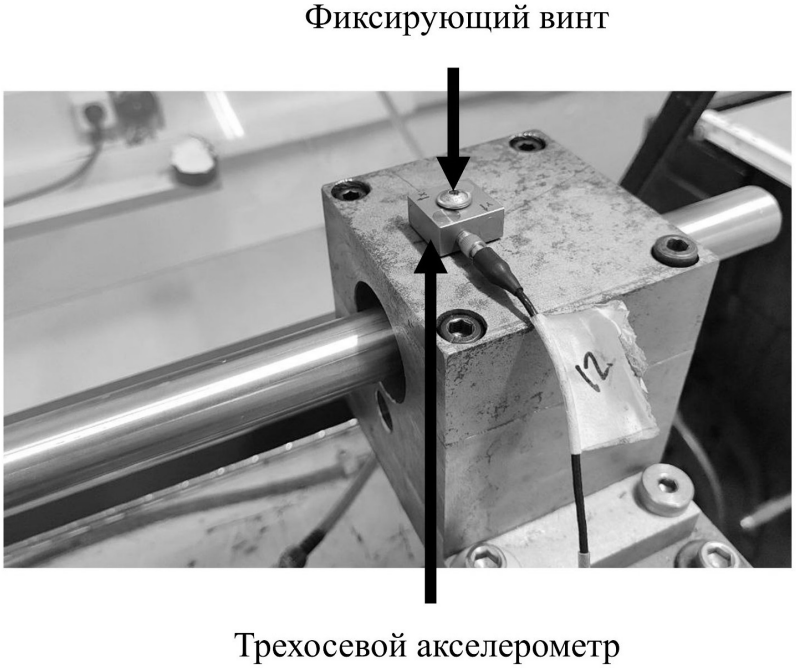

Рис. 2. Опора с закрепленным на ней акселерометром

В ходе эксперимента были записаны четыре типа сигналов, соответствующих следующим состояниям подшипников: исправный подшипник, исправный подшипник с ослабленным креплением датчика, подшипник с дефектом внешнего кольца и подшипник с дефектом внешнего кольца и ослабленным креплением датчика. Имитация плохого закрепления осуществлялась путем ослабления фиксирующего винта на четверть оборота.

Сначала проводилась серия экспериментов с исправным подшипником, при этом поочередно регистрировались сигналы как для нормального, так и для ослабленного крепления датчика. Из-за небольшой перестановки датчика после каждого ослабления отдельные серии экспериментов не являются точными повторениями, что вносит разнообразие в данные. Аналогично проводилась серия экспериментов с подшипником, имеющим дефект внешнего кольца. Частота дискретизации сигналов во всех экспериментах составляла 100 кГц. Первые четыре фрагмента каждого эксперимента записывались в течение 2.5 минут, пятый фрагмент — 2 минуты. Таким образом, суммарная длительность сигналов для каждого класса составила 12 минут.

На рис. 3 показано, что во временной области сигналы для нормального и ослабленного крепления датчика практически неразличимы, однако при переходе к амплитудному спектру различия становятся заметными. Эти изменения спектра, вызванные неисправностью измерительного средства, могут привести к ложным срабатываниям диагностической модели, что особенно важно учитывать при разработке систем технического мониторинга.

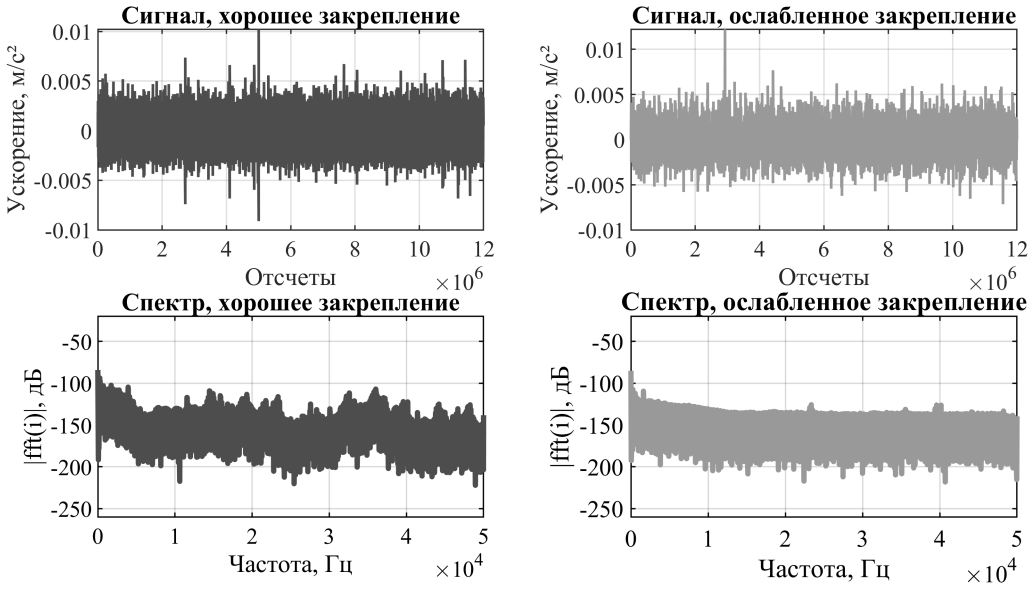

Рис. 3. Сигналы и их спектры для исправного подшипника при хорошем и ослабленном креплении датчика

2.2. Предобработка и выделение LPC-признаков из экспериментальных данных

На первом этапе обработки исходные временные сигналы были разбиты на короткие фрагменты фиксированной длины. В соответствии с результатами, представленными в работе [30], продолжительность фрагмента в 0.2 секунды оказывается достаточной для решения задач диагностики состояния подшипников. Для увеличения объема выборки применялось скользящее окно с перекрытием 0.1 секунды. Этот подход не только расширяет количество обучающих примеров, но и сохраняет временную зависимость между соседними фрагментами, что позволяет учитывать контекст изменения сигнала и снижает вероятность ложных срабатываний диагностической модели.

При частоте дискретизации 100 кГц каждый фрагмент содержал 20000 отсчетов, а шаг смещения окна составлял 10000 отсчетов. В результате сегментации для каждого из четырех классов было получено по 7195 временных рядов.

После сегментации для каждого временного ряда вычислялся спектр Фурье в диапазоне частот от 500 Гц до 40000 Гц. Такой диапазон исключает низкочастотные компоненты, напрямую связанные с частотой вращения подшипника, и включает резонансную частоту датчика, что делает его достаточным для выявления как дефектов подшипника, так и неисправностей измерительной системы.

Для выбора порядка p модели LPC использовались три стандартные функции потерь:

$$\mathrm{FPE}(p) = \frac{N + (p+1)}{N - (p+1)}\hat{\sigma}^2, \tag{13}$$

$$\mathrm{AIC}(p) = N \ln \hat{\sigma}^2 + 2p, \tag{14}$$

$$\mathrm{MDL}(p) = N \ln \hat{\sigma}^2 + p \ln N, \tag{15}$$

где N — количество элементов временного ряда, а $\hat{\sigma}^2$ — оценка дисперсии. Результаты расчета функций потерь для полученных сигналов представлены на рис. 4.

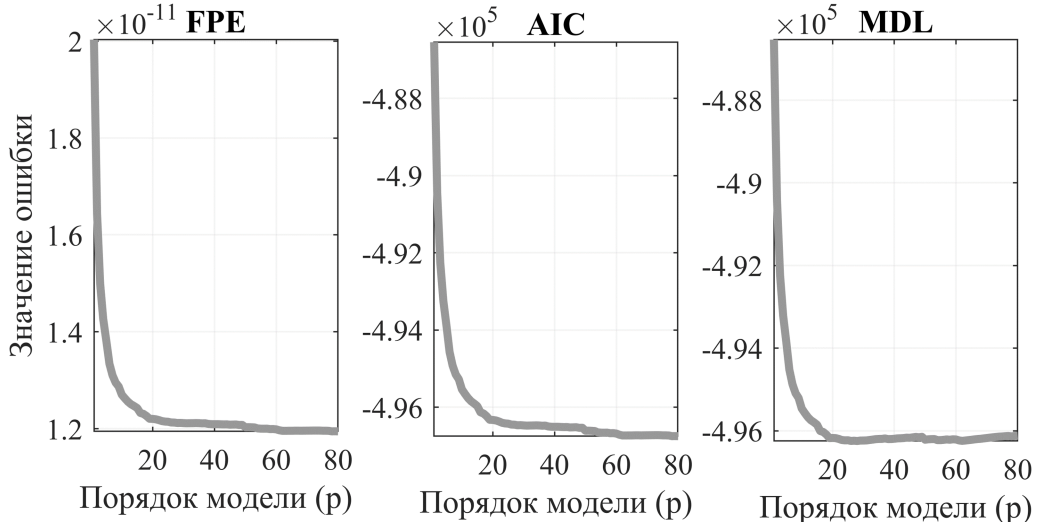

Рис. 4. Зависимость функций потерь (FPE, AIC и MDL) от порядка модели LPC

Как видно из рис. 4, все кривые функций потерь сначала быстро убывают при увеличении порядка модели, однако после определенного значения их снижение практически прекращается. Это соответствует классическому правилу «локтя»: порядок модели выбирается в точке, после которой дальнейшее увеличение числа коэффициентов не приводит к существенному улучшению качества восстановления. В рамках данной работы в качестве компромиссного решения между точностью моделирования и вычислительной эффективностью выбран порядок $p = 50$. На рис. 5 представлены средние значения LPC-коэффициентов для всех четырех рассматриваемых случаев, визуализированные в виде изображений в соответствии с матрицей (11).

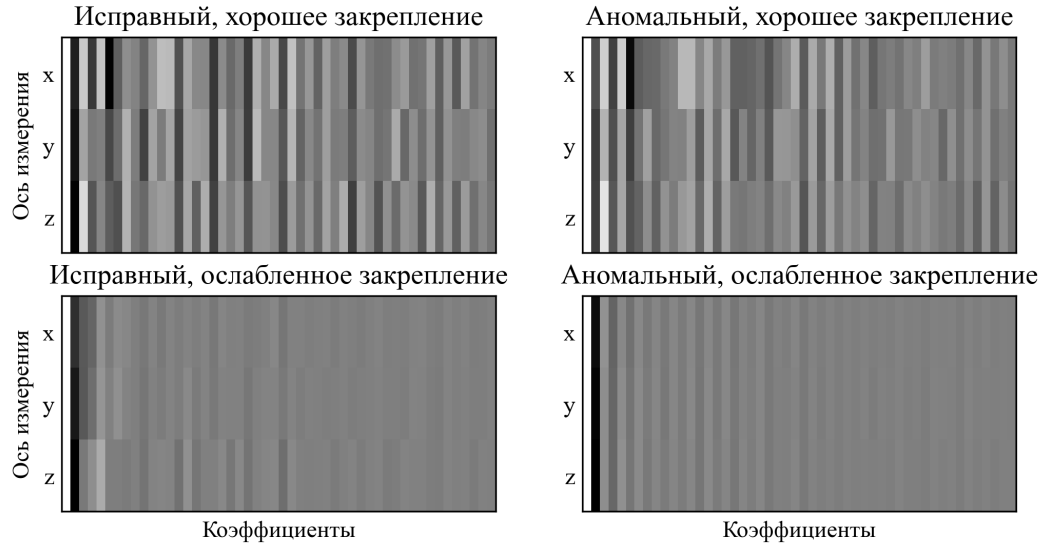

Рис. 5. Средние значения матриц коэффициентов для каждого из 4 возможных случаев, представленные в виде изображений

Для анализа разделимости классов и визуализации полученных признаков 150-мерные векторы (полученные из матриц LPC-коэффициентов размером 3×50) были спроецированы в двумерное пространство с помощью метода PaCMAP [32], рис. 6. Этот метод сочетает преимущества нелинейного уменьшения размерности, сохраняя как глобальную структуру данных, так и локальные взаимосвязи между точками, что особенно важно для задач классификации.

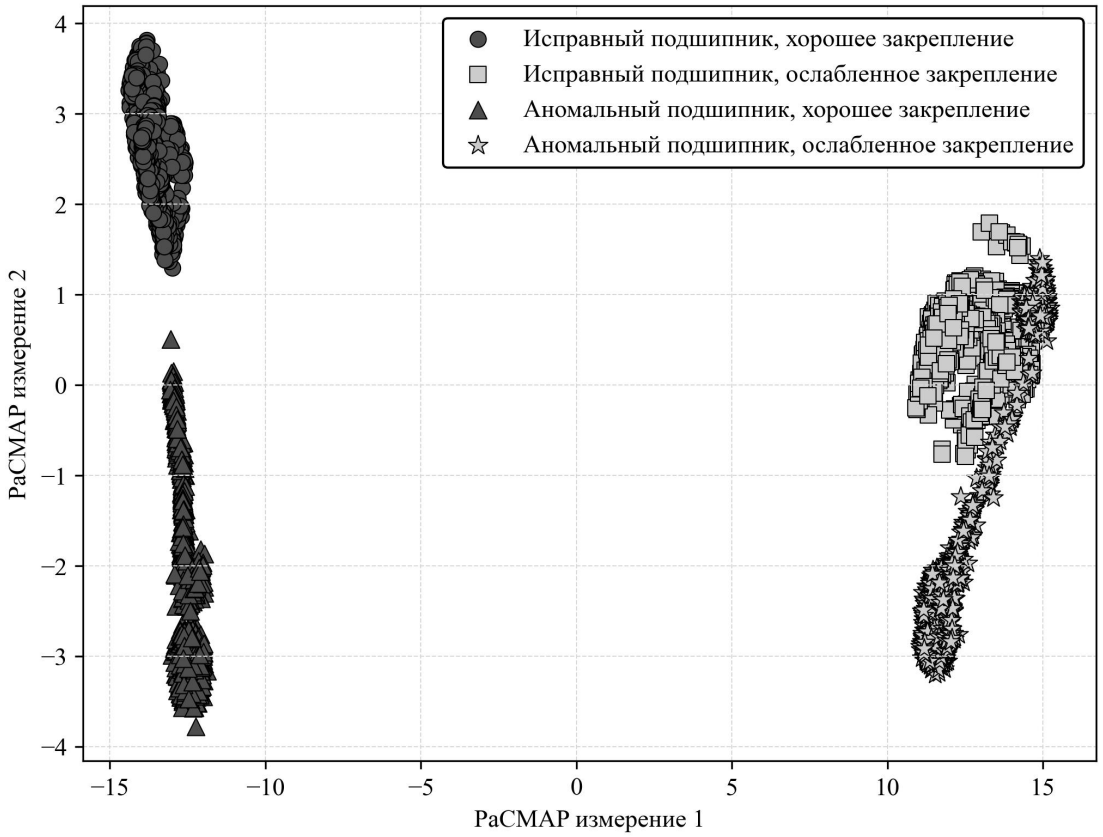

Рис. 6. Отображение полученных после предобработки данных в двухмерном пространстве с использованием метода PaCMAP

Анализ распределения признаков после снижения размерности демонстрирует четкое разделение данных для исправного и дефектного подшипника при нормальном закреплении датчика, а также выраженные различия между сигналами при нормальном и ослабленном креплении акселерометра.

При этом наблюдается частичное перекрытие кластеров, соответствующих дефектному и исправному подшипнику в случае ослабленного закрепления. Это указывает на то, что в таких условиях признаки сигналов становятся сходными, что может привести к тому, что модель будет классифицировать их как один и тот же класс данных.

3. Параметры и тестирование каскадной модели C-LPC-AE

3.1. Параметры автоэнкодеров

Предлагаемая в данной работе каскадная модель строилась с использованием двух идентичных сверточных автоэнкодеров, обучаемых на двух различных классах состояний подшипника. Первый автоэнкодер обучался на данных, полученных с исправного подшип-

ника при корректном закреплении акселерометра, тогда как второй автоэнкодер использовался для обработки сигналов, полученных при ослабленном креплении датчика.

Архитектура каждого автоэнкодера включает энкодер и декодер, объединенные симметричной структурой. На вход подаются двумерные массивы признаков размером 3×50, где три строки соответствуют измеряемым каналам акселерометра, а 50 столбцов — LPC-коэффициентам. Такая организация данных сохраняет пространственные взаимосвязи между коэффициентами одного порядка.

Энкодер состоит из двух сверточных слоев (Conv2D с ядром 3×3 и активацией ReLU) с промежуточными слоями субдискретизации (MaxPooling2D с окном 2×1), что обеспечивает последовательное сжатие пространственных признаков и выделение устойчивых паттернов. После сверточных слоев данные переводятся в одномерное представление (Flatten) и проецируются в латентное пространство размерности 8 с помощью полносвязного слоя Dense.

Декодер выполняет обратное преобразование: данные проходят через полносвязный слой, восстанавливающий пространственную форму (Reshape), затем через два блока увеличения размерности (UpSampling2D) и сверточные слои с активацией ReLU. Для коррекции размеров применяется слой Cropping2D, а финальный сверточный слой с линейной активацией формирует выходное изображение, соответствующее реконструированным входным данным. В качестве функции ошибки применялась среднеквадратичная ошибка MSE (Mean Squared Error):

$$MSE = \frac{1}{M \cdot p} \sum_{i=1}^{M} \sum_{n=1}^{p} \left(\hat{x}_n^{(i)} - x_n^{(i)} \right)^2, \tag{16}$$

где $x_n^{(i)}$ — значение n-го коэффициента для i-го канала, $i = 1, \ldots, M$, $\hat{x}_n^{(i)}$ — соответствующий восстановленный отсчет. (В нашем случае, $M = 3$, $p = 50$.)

Модели 1 и 2, соответствующие классу исправного подшипника и исправного подшипника с ослабленным креплением датчика вибрации, обучались с идентичными гиперпараметрами: 50 эпох, размер батча 64, оптимизатор Adam с шагом обучения 10^{-4}, выбранными на основе серии предварительных экспериментов. Для обучения использовались 10 минут экспериментальных данных, соответствующих 5996 образцам. Окно сглаживания соответствовало 0.5 секундам.

3.2. Тестирование каскадной модели C-LPC-AE

Для оценки обобщающей способности предложенной каскадной модели автоэнкодеров тестирование проводилось на отложенных выборках. Первая ступень каскада представляет собой автоэнкодер, обученный на данных с исправного подшипника с корректным закреплением акселерометра (10 минут, 5996 образцов). На рис. 7 представлены значения его ошибки реконструкции для различных типов сигналов. Для исправного случая с хорошим закреплением конец тренировочных данных обозначен штрих-пунктирной линией. Как видно, ошибка реконструкции на тестовых данных несколько выше, чем на обучающих, однако остается ниже установленного порогового значения, за исключением единичного случая ложного срабатывания.

Для остальных трех типов данных — аномальный подшипник с хорошим закреплением датчика, исправный подшипник с ослабленным креплением акселерометра и аномальный подшипник с ослабленным закреплением — ошибка реконструкции вычислялась на протяжении всех 12 минут сигналов и, как можно видеть на рис. 7, во всех случаях превышала по-

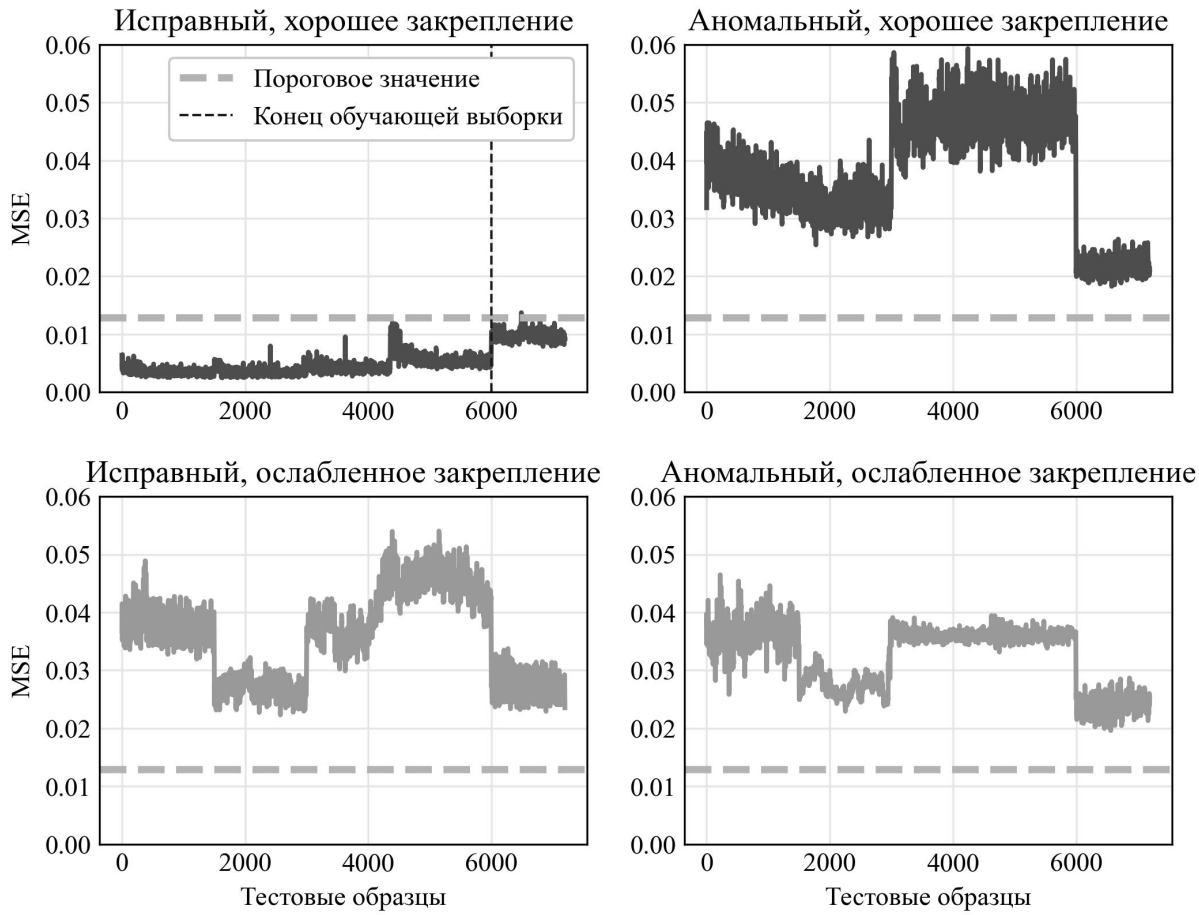

Рис. 7. Ошибка реконструкции первой модели автоэнкодера для разных типов данных

роговое значение. Это демонстрирует способность первой ступени каскадной модели выявлять отклонения от нормального состояния подшипника. Однако данный уровень реакции модели не позволяет различить, являются ли отклонения следствием аномалии подшипника или ослабления крепления датчика. Для этого необходима вторая ступень каскадной модели C-LPC-AE, предназначенная для дифференциации источников этих аномалий.

Результаты тестирования второго автоэнкодера, обученного на данных исправного подшипника с ослабленным креплением датчика, представлены на рис. 8. Аналогично первой модели, конец тренировочных данных обозначен штрих-пунктирной линией. Следует отметить, что в тестировании не использовались данные исправного подшипника с корректным закреплением, поскольку данный автоэнкодер предназначен для обработки сигналов только в случае выявления отклонений на первой ступени каскадной модели.

Обученная модель демонстрирует высокие значения ошибки реконструкции для аномального подшипника с корректным закреплением датчика, что позволяет уверенно выявлять этот тип отклонения.

В случае аномального подшипника с ослабленным креплением датчика наблюдается смешение сигналов, и часть срабатываний модели может интерпретироваться как неисправность крепления, а часть — как реальная аномалия подшипника. Это объясняется высокой схожестью характеристик сигналов при ослабленном креплении, что затрудняет их однозначное разделение, как было показано на рис. 6. На рис. 8 (нижний график) 37% срабатываний указывают на ослабление крепления датчика, а 63% — на наличие аномалии

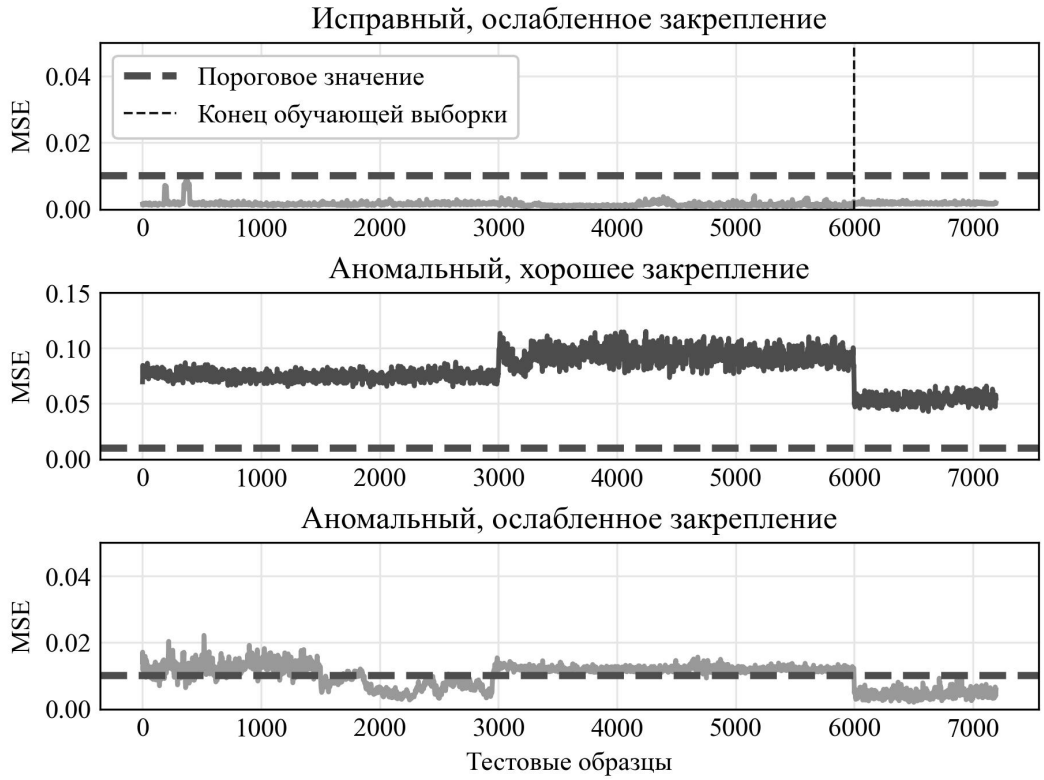

Рис. 8. Ошибка реконструкции второй модели автоэнкодера для разных типов данных

подшипника. Такой результат является удовлетворительным: при выявлении проблемы с креплением датчика достаточно устранить ее механически, после чего модель сможет корректно зафиксировать реальное аномальное состояние подшипника, что подтверждается высокой ошибкой реконструкции для данных с аномальным подшипником при правильной установке сенсора.

Таким образом, каскадная модель C-LPC-AE эффективно различает основные типы отклонений: нормальное состояние подшипника, аномалии в подшипнике и артефакты, вызванные ослаблением крепления датчика. В случае аномального подшипника с ослабленным креплением наблюдается частичное смешение сигналов: часть отклонений может интерпретироваться как проблема с креплением, а часть — как реальная неисправность подшипника, что отражает реальную сложность задачи. В практических условиях такие сигналы можно выделять в отдельную «зону неопределенности», требующую дополнительной проверки. При устранении механических проблем с креплением датчика сигналы перестают содержать искажения, вызванные некорректной установкой, и любые превышения порога реконструкции второго автоэнкодера теперь могут быть надежно интерпретированы как реальные аномалии подшипника.

Заключение

В данной работе предложен метод обеспечения достоверности измерений в системах технического мониторинга на основе каскадной архитектуры сверточных автоэнкодеров C-LPC-AE. Подход сочетает эффективное извлечение информативных признаков на основе линейного предсказательного кодирования и двухступенчатое обнаружение аномалий с использованием сверточных автоэнкодеров, что позволяет не только выявлять дефекты

оборудования, но и дифференцировать их от артефактов, вызванных нарушениями в работе измерительной системы.

Ключевой особенностью предложенной архитектуры является ее обучение на нормальных и нештатных состояниях измерительного узла — в частности, на сигналах с ослабленным креплением датчика. Это устраняет необходимость сбора данных о реальных отказах, которые редки, труднодоступны и часто не могут быть получены в контролируемых условиях.

Экспериментальные результаты подтвердили высокую чувствительность модели к изменениям в состоянии подшипника и способность автоматически фильтровать ложные тревоги, вызванные неисправностями сенсоров. Использование LPC-признаков обеспечивает компактное и информативное представление сигнала, снижая вычислительную нагрузку и упрощая интеграцию модели в промышленные системы реального времени. Предложенный подход повышает надежность и автономность мониторинга, что особенно важно для цифровой индустрии.

Исследование выполнено при финансовой поддержке Министерства науки и высшего образования Российской Федерации (государственное задание на выполнение фундаментальных научных исследований № FENU-2023-0010 (2023010ГЗ)).

Литература

1. Randall R., Smith W. Detection of faulty accelerometer mounting from response measurements // Journal of Sound and Vibration. 2020. Vol. 477. P. 115318. DOI: 10.1016/j.jsv.2020.115318.
2. Abboud D., Elbadaoui M., Becquerelle S., Lalmi M. Detection of Sensor Detachment in Aircraft Engines Using Vibration Signals // Proceedings of the 10th International Conference on Rotor Dynamics – IFToMM / ed. by K.L. Cavalca, H.I. Weber. Cham: Springer International Publishing, 2019. P. 351–365. DOI: 10.1007/978-3-319-99268-6_25.
3. Peng B., Bi Y., Xue B., *et al.* A Survey on Fault Diagnosis of Rolling Bearings // Algorithms. 2022. Vol. 15, no. 10. DOI: 10.3390/a15100347.
4. Kannan V., Zhang T., Li H. A Review of the Intelligent Condition Monitoring of Rolling Element Bearings // Machines. 2024. Vol. 12, no. 7. DOI: 10.3390/machines12070484.
5. Wu G., Yan T., Yang G., *et al.* A Review on Rolling Bearing Fault Signal Detection Methods Based on Different Sensors // Sensors. 2022. Vol. 22, no. 21. DOI: 10.3390/s22218330.
6. Li L., Qu L. Cyclic statistics in rolling bearing diagnosis // Journal of Sound and Vibration. 2003. Vol. 267, no. 2. P. 253–265. DOI: 10.1016/S0022-460X(02)01412-8.
7. Karacay T., Akturk N. Experimental diagnostics of ball bearings using statistical and spectral methods // Tribology International. 2009. Vol. 42, no. 6. P. 836–843. DOI: 10.1016/j.triboint.2008.11.003.
8. Gupta P., Pradhan M. Fault detection analysis in rolling element bearing: A review // Materials Today: Proceedings. 2017. Vol. 4, no. 2, Part A. P. 2085–2094. 5th International Conference of Materials Processing and Characterization (ICMPC 2016). DOI: 10.1016/j.matpr.2017.02.054.

9. Gao C., Liu S., Zhang X. A new method of adaptive Fourier modal decomposition and its application to rolling bearing fault diagnosis // Structural Health Monitoring. 2025. DOI: 10.1177/14759217251347534.

10. Li D.Z., Wang W., Ismail F. An Enhanced Bispectrum Technique With Auxiliary Frequency Injection for Induction Motor Health Condition Monitoring // IEEE Transactions on Instrumentation and Measurement. 2015. Vol. 64, no. 10. P. 2679–2687. DOI: 10.1109/TIM.2015.2419031.

11. Zhen L., Zhengjia H., Yanyang Z., Xuefeng C. Bearing condition monitoring based on shock pulse method and improved redundant lifting scheme // Mathematics and Computers in Simulation. 2008. Vol. 79, no. 3. P. 318–338. DOI: 10.1016/j.matcom.2007.12.004.

12. Butler D. The Shock-pulse method for the detection of damaged rolling bearings // Non-Destructive Testing. 1973. Vol. 6, no. 2. P. 92–95. DOI: 10.1016/0029-1021(73)90116-3.

13. He Y., Hu M., Feng K., Jiang Z. Bearing Condition Evaluation Based on the Shock Pulse Method and Principal Resonance Analysis // IEEE Transactions on Instrumentation and Measurement. 2021. Vol. 70. P. 1–12. DOI: 10.1109/TIM.2021.3050679.

14. Zhang Q., Deng L. An Intelligent Fault Diagnosis Method of Rolling Bearings Based on Short-Time Fourier Transform and Convolutional Neural Network // Journal of Failure Analysis and Prevention. 2023. Vol. 23. P. 795–811. DOI: 10.1007/s11668-023-01616-9.

15. Yue Y., Wang H., Zhang S. Mel frequency mapping for intelligent diagnosis of rolling element bearings across different working conditions // Applied Acoustics. 2024. Vol. 220. P. 109944. DOI: 10.1016/j.apacoust.2024.109944.

16. Xie F., Li G., Song C., Song M. The Early Diagnosis of Rolling Bearings' Faults Using Fractional Fourier Transform Information Fusion and a Lightweight Neural Network // Fractal and Fractional. 2023. Vol. 7, no. 12. DOI: 10.3390/fractalfract7120875.

17. Qin Y., Yang R., Shi H., *et al.* Adaptive Fast Chirplet Transform and Its Application Into Rolling Bearing Fault Diagnosis Under Time-Varying Speed Condition // IEEE Transactions on Instrumentation and Measurement. 2023. Vol. 72. P. 1–12. DOI: 10.1109/TIM.2023.3282660.

18. Raj K.K., Kumar S., Kumar R.R. Systematic Review of Bearing Component Failure: Strategies for Diagnosis and Prognosis in Rotating Machinery // Arabian Journal for Science and Engineering. 2024. Vol. 50. P. 5353–5375. DOI: 10.1007/s13369-024-09866-x.

19. Yuan Y., Wei J., Huang H., *et al.* Review of resampling techniques for the treatment of imbalanced industrial data classification in equipment condition monitoring // Engineering Applications of Artificial Intelligence. 2023. Vol. 126. P. 106911. DOI: https://doi.org/10.1016/j.engappai.2023.106911.

20. Liang P., Yu Z., Wang B., *et al.* Fault transfer diagnosis of rolling bearings across multiple working conditions via subdomain adaptation and improved vision transformer network // Advanced Engineering Informatics. 2023. Vol. 57. P. 102075. DOI: 10.1016/j.aei.2023.102075.

21. Liu S., Jiang H., Wu Z., Li X. Data synthesis using deep feature enhanced generative adversarial networks for rolling bearing imbalanced fault diagnosis // Mechanical Systems and Signal Processing. 2022. Vol. 163. P. 108139. DOI: 10.1016/j.ymssp.2021.108139.

22. Iglesias G., Talavera E., González-Prieto Á., *et al.* Data Augmentation techniques in time series domain: a survey and taxonomy // Neural Computing and Applications. 2023. Vol. 35. P. 10123–10145. DOI: 10.1007/s00521-023-08459-3.

23. Lebedev D.K. Sensitivity of the Rolling Bearings Diagnostic Method Depending on the Number of Measuring Points of a Multipoint Temperature Sensor // 2025 27th International Conference on Digital Signal Processing and its Applications (DSPA). 2025. P. 1–5. DOI: 10.1109/DSPA64310.2025.10977932.

24. Гоглачев А.И. Классификация потокового временного ряда на основе нейросетевых технологий и поведенческих шаблонов // Вестник ЮУрГУ. Серия: Вычислительная математика и информатика. 2024. Т. 13, № 3. С. 79–94. DOI: 10.14529/cmse240305.

25. Краева Я.А. Обнаружение аномалий временного ряда на основе технологий интеллектуального анализа данных и нейронных сетей // Вестник ЮУрГУ. Серия: Вычислительная математика и информатика. 2023. Т. 12, № 3. С. 50–71. DOI: 10.14529/cmse230304.

26. Givnan S., Chalmers C., Fergus P., *et al.* Anomaly Detection Using Autoencoder Reconstruction upon Industrial Motors // Sensors. 2022. Vol. 22, no. 9. DOI: 10.3390/s22093166.

27. Ahmad S., Styp-Rekowski K., Nedelkoski S., Kao O. Autoencoder-based Condition Monitoring and Anomaly Detection Method for Rotating Machines // 2020 IEEE International Conference on Big Data (Big Data). 2020. P. 4093–4102. DOI: 10.1109/BigData50022.2020.9378015.

28. Malviya V., Mukherjee I., Tallur S. Edge-Compatible Convolutional Autoencoder Implemented on FPGA for Anomaly Detection in Vibration Condition-Based Monitoring // IEEE Sensors Letters. 2022. Vol. 6, no. 4. P. 1–4. DOI: 10.1109/LSENS.2022.3159972.

29. Shestakov A.L., Lebedev D.K., Sinitsin V.V., *et al.* Intelligent Multipoint Temperature Sensors Data Processing for Rolling Bearings Diagnosis // 2024 XXXIV International Scientific Symposium Metrology and Metrology Assurance (MMA). 2024. P. 1–6. DOI: 10.1109/MMA62616.2024.10817658.

30. Mohammad M., Ibryaeva O., Sinitsin V., Eremeeva V. A Computationally Efficient Method for the Diagnosis of Defects in Rolling Bearings Based on Linear Predictive Coding // Algorithms. 2025. Vol. 18, no. 2. DOI: 10.3390/a18020058.

31. Thimmaraja Y.G., Nagaraja B.G., Jayanna H.S. Speech enhancement and encoding by combining SS-VAD and LPC // International Journal of Speech Technology. 2021. Vol. 24. P. 165–172. DOI: 10.1007/s10772-020-09786-9.

32. Wang Y., Huang H., Rudin C., Shaposhnik Y. Understanding How Dimension Reduction Tools Work: An Empirical Approach to Deciphering t-SNE, UMAP, TriMap, and PaCMAP for Data Visualization // Journal of Machine Learning Research. 2021. Vol. 22, no. 201. P. 1–73. URL: http://jmlr.org/papers/v22/20-1061.html.

Галышев Дмитрий Вячеславович, студент кафедры «Прикладная математика и программирование», Южно-Уральский государственный университет (национальный исследовательский университет) (Челябинск, Российская Федерация)

Яковенко Артем Дмитриевич, аспирант, Южно-Уральский государственный университет (национальный исследовательский университет) (Челябинск, Российская Федерация)

Ибряева Ольга Леонидовна, канд. физ.-мат. наук, доцент, старший научный сотрудник «НИЛ технической самодиагностики и самоконтроля приборов и систем», Южно-Уральский государственный университет (национальный исследовательский университет) (Челябинск, Российская Федерация)

Шестаков Александр Леонидович, докт. техн. наук, проф., заведующий «НИЛ технической самодиагностики и самоконтроля приборов и систем», Южно-Уральский государственный университет (национальный исследовательский университет) (Челябинск, Российская Федерация)

DOI: 10.14529/cmse250401

CONTROL OF MEASUREMENT RELIABILITY IN TECHNICAL MONITORING INSTRUMENTS USING A CASCADE OF AUTOENCODERS

© 2025 D.V. Galyshev, A.D. Yakovenko, O.L. Ibrayeva, A.L. Shestakov

South Ural State University (pr. Lenina 76, Chelyabinsk, 454080 Russia)

E-mail: galyshevdv@susu.ru, iakovenkoad@susu.ru, ibriaevaol@susu.ru, a.l.shestakov@susu.ru

Received: 17.08.2025

The paper proposes a method for ensuring measurement reliability in technical monitoring systems – a cascaded model C-LPC-AE that combines informative spectral feature extraction based on Linear Predictive Coding (LPC) with a two-stage architecture of convolutional autoencoders. The method is designed for bearing condition diagnostics and simultaneous verification of sensor operational integrity, which is particularly relevant in digital industry environments requiring high autonomy and reliability of monitoring systems. The first stage of the cascade, trained on signals from a healthy bearing with properly mounted sensors, performs anomaly detection based on reconstruction error. The second stage, trained on data with a loosened accelerometer mount, analyzes the nature of the anomaly and enables differentiation between bearing faults and signal distortions caused by improper sensor installation. A key advantage of the approach is that it does not require data from actual equipment failures: training is performed exclusively on easily reproducible conditions — normal operation and simulated sensor mounting faults. Experiments were conducted using data from the SpectraQuest test rig, including bearings with an artificially introduced outer race defect. The results demonstrate high model sensitivity to actual bearing defects and sensor mounting issues. The use of LPC-based features ensures compact signal representation and reduces computational load, making the proposed approach promising for integration into real-time industrial diagnostic systems.

Keywords: vibration diagnostics, bearing diagnostics, loosened sensor mounting, autoencoder, anomaly detection, technical monitoring, linear predictive coding, spectral analysis, measurement reliability.

FOR CITATION

Galyshev D.V., Yakovenko A.D., Ibrayeva O.L., Shestakov A.L. Control of Measurement Reliability in Technical Monitoring Instruments Using a Cascade of Autoencoders. Bulletin of the South Ural State University. Series: Computational Mathematics and Software Engineering. 2025. Vol. 14, no. 4. P. 5–24. (in Russian) DOI: 10.14529/cmse250401.

References

1. Randall R., Smith W. Detection of faulty accelerometer mounting from response measurements. Journal of Sound and Vibration. 2020. Vol. 477. P. 115318. DOI: 10.1016/j.jsv.2020.115318.
2. Abboud D., Elbadaoui M., Becquerelle S., Lalmi M. Detection of Sensor Detachment in Aircraft Engines Using Vibration Signals. Proceedings of the 10th International Conference on Rotor Dynamics – IFToMM / ed. by K.L. Cavalca, H.I. Weber. Cham: Springer International Publishing, 2019. P. 351–365. DOI: 10.1007/978-3-319-99268-6_25.
3. Peng B., Bi Y., Xue B., *et al.* A Survey on Fault Diagnosis of Rolling Bearings. Algorithms. 2022. Vol. 15, no. 10. DOI: 10.3390/a15100347.
4. Kannan V., Zhang T., Li H. A Review of the Intelligent Condition Monitoring of Rolling Element Bearings. Machines. 2024. Vol. 12, no. 7. DOI: 10.3390/machines12070484.
5. Wu G., Yan T., Yang G., *et al.* A Review on Rolling Bearing Fault Signal Detection Methods Based on Different Sensors. Sensors. 2022. Vol. 22, no. 21. DOI: 10.3390/s22218330.
6. Li L., Qu L. Cyclic statistics in rolling bearing diagnosis. Journal of Sound and Vibration. 2003. Vol. 267, no. 2. P. 253–265. DOI: 10.1016/S0022-460X(02)01412-8.
7. Karacay T., Akturk N. Experimental diagnostics of ball bearings using statistical and spectral methods. Tribology International. 2009. Vol. 42, no. 6. P. 836–843. DOI: 10.1016/j.triboint.2008.11.003.
8. Gupta P., Pradhan M. Fault detection analysis in rolling element bearing: A review. Materials Today: Proceedings. 2017. Vol. 4, no. 2, Part A. P. 2085–2094. 5th International Conference of Materials Processing and Characterization (ICMPC 2016). DOI: 10.1016/j.matpr.2017.02.054.
9. Gao C., Liu S., Zhang X. A new method of adaptive Fourier modal decomposition and its application to rolling bearing fault diagnosis. Structural Health Monitoring. 2025. DOI: 10.1177/14759217251347534.
10. Li D.Z., Wang W., Ismail F. An Enhanced Bispectrum Technique With Auxiliary Frequency Injection for Induction Motor Health Condition Monitoring. IEEE Transactions on Instrumentation and Measurement. 2015. Vol. 64, no. 10. P. 2679–2687. DOI: 10.1109/TIM.2015.2419031.
11. Zhen L., Zhengjia H., Yanyang Z., Xuefeng C. Bearing condition monitoring based on shock pulse method and improved redundant lifting scheme. Mathematics and Computers in Simulation. 2008. Vol. 79, no. 3. P. 318–338. DOI: 10.1016/j.matcom.2007.12.004.
12. Butler D. The Shock-pulse method for the detection of damaged rolling bearings. Non-Destructive Testing. 1973. Vol. 6, no. 2. P. 92–95. DOI: 10.1016/0029-1021(73)90116-3.
13. He Y., Hu M., Feng K., Jiang Z. Bearing Condition Evaluation Based on the Shock Pulse Method and Principal Resonance Analysis. IEEE Transactions on Instrumentation and Measurement. 2021. Vol. 70. P. 1–12. DOI: 10.1109/TIM.2021.3050679.
14. Zhang Q., Deng L. An Intelligent Fault Diagnosis Method of Rolling Bearings Based on Short-Time Fourier Transform and Convolutional Neural Network. Journal of Failure Analysis and Prevention. 2023. Vol. 23. P. 795–811. DOI: 10.1007/s11668-023-01616-9.

15. Yue Y., Wang H., Zhang S. Mel frequency mapping for intelligent diagnosis of rolling element bearings across different working conditions. Applied Acoustics. 2024. Vol. 220. P. 109944. DOI: 10.1016/j.apacoust.2024.109944.

16. Xie F., Li G., Song C., Song M. The Early Diagnosis of Rolling Bearings' Faults Using Fractional Fourier Transform Information Fusion and a Lightweight Neural Network. Fractal and Fractional. 2023. Vol. 7, no. 12. DOI: 10.3390/fractalfract7120875.

17. Qin Y., Yang R., Shi H., *et al.* Adaptive Fast Chirplet Transform and Its Application Into Rolling Bearing Fault Diagnosis Under Time-Varying Speed Condition. IEEE Transactions on Instrumentation and Measurement. 2023. Vol. 72. P. 1–12. DOI: 10.1109/TIM.2023.3282660.

18. Raj K.K., Kumar S., Kumar R.R. Systematic Review of Bearing Component Failure: Strategies for Diagnosis and Prognosis in Rotating Machinery. rabian Journal for Science and Engineering. 2024. Vol. 50. P. 5353–5375. DOI: 10.1007/s13369-024-09866-x.

19. Yuan Y., Wei J., Huang H., *et al.* Review of resampling techniques for the treatment of imbalanced industrial data classification in equipment condition monitoring. Engineering Applications of Artificial Intelligence. 2023. Vol. 126. P. 106911. DOI: 10.1016/j.engappai.2023.106911.

20. Liang P., Yu Z., Wang B., *et al.* Fault transfer diagnosis of rolling bearings across multiple working conditions via subdomain adaptation and improved vision transformer network. Advanced Engineering Informatics. 2023. Vol. 57. P. 102075. DOI: 10.1016/j.aei.2023.102075.

21. Liu S., Jiang H., Wu Z., Li X. Data synthesis using deep feature enhanced generative adversarial networks for rolling bearing imbalanced fault diagnosis. Mechanical Systems and Signal Processing. 2022. Vol. 163. P. 108139. DOI: 10.1016/j.ymssp.2021.108139.

22. Iglesias G., Talavera E., González-Prieto Á., *et al.* Data Augmentation techniques in time series domain: a survey and taxonomy. Neural Computing and Applications. 2023. Vol. 35. P. 10123–10145. DOI: 10.1007/s00521-023-08459-3.

23. Lebedev D.K. Sensitivity of the Rolling Bearings Diagnostic Method Depending on the Number of Measuring Points of a Multipoint Temperature Sensor. 2025 27th International Conference on Digital Signal Processing and its Applications (DSPA). 2025. P. 1–5. DOI: 10.1109/DSPA64310.2025.10977932.

24. Goglachev A.I. Classification of Streaming Time Series Based on Neural Network Technologies and Behavioral Patterns. Bulletin of the South Ural State University. Series: Computational Mathematics and Software Engineering. 2024. Vol. 13, no. 3. P. 79–94. DOI: 10.14529/cmse240305.

25. Kraeva Y.A. Anomaly Detection in Time Series Based on Data Mining and Neural Network Technologies. Bulletin of the South Ural State University. Series: Computational Mathematics and Software Engineering. 2023. Vol. 12, no. 3. P. 50–71. DOI: 10.14529/cmse230304.

26. Givnan S., Chalmers C., Fergus P., *et al.* Anomaly Detection Using Autoencoder Reconstruction upon Industrial Motors. Sensors. 2022. Vol. 22, no. 9. DOI: 10.3390/s22093166.

27. Ahmad S., Styp-Rekowski K., Nedelkoski S., Kao O. Autoencoder-based Condition Monitoring and Anomaly Detection Method for Rotating Machines. 2020 IEEE International Conference on Big Data (Big Data). 2020. P. 4093–4102. DOI: 10.1109/BigData50022.2020.9378015.

28. Malviya V., Mukherjee I., Tallur S. Edge-Compatible Convolutional Autoencoder Implemented on FPGA for Anomaly Detection in Vibration Condition-Based Monitoring. IEEE Sensors Letters. 2022. Vol. 6, no. 4. P. 1–4. DOI: 10.1109/LSENS.2022.3159972.

29. Shestakov A.L., Lebedev D.K., Sinitsin V.V., *et al.* Intelligent Multipoint Temperature Sensors Data Processing for Rolling Bearings Diagnosis. 2024 XXXIV International Scientific Symposium Metrology and Metrology Assurance (MMA). 2024. P. 1–6. DOI: 10.1109/MMA62616.2024.10817658.

30. Mohammad M., Ibryaeva O., Sinitsin V., Eremeeva V. A Computationally Efficient Method for the Diagnosis of Defects in Rolling Bearings Based on Linear Predictive Coding. Algorithms. 2025. Vol. 18, no. 2. DOI: 10.3390/a18020058.

31. Thimmaraja Y.G., Nagaraja B.G., Jayanna H.S. Speech enhancement and encoding by combining SS-VAD and LPC. International Journal of Speech Technology. 2021. Vol. 24. P. 165–172. DOI: 10.1007/s10772-020-09786-9.

32. Wang Y., Huang H., Rudin C., Shaposhnik Y. Understanding How Dimension Reduction Tools Work: An Empirical Approach to Deciphering t-SNE, UMAP, TriMap, and PaCMAP for Data Visualization. Journal of Machine Learning Research. 2021. Vol. 22, no. 201. P. 1–73. URL: http://jmlr.org/papers/v22/20-1061.html.

УДК 004.652 **DOI: 10.14529/cmse250402**

СИНХРОНИЗАЦИЯ ДАННЫХ МЕЖДУ ТАБЛИЦАМИ СПЕЦИАЛЬНОГО ВИДА И БАЗОЙ ДАННЫХ

© 2025 С.В. Зыкин[1], В.С. Зыкин[1], Н.С. Шепелев[2]

[1]*Институт математики им. С.Л. Соболева СО РАН*
(630090 Новосибирск, пр. ак. Коптюга, д. 4),
[2]*Омский государственный технический университет*
(644050 Омск, пр. Мира, д. 11)
E-mail: zykin@ofim.oscsbras.ru, vszykin@mail.ru, n06k@mail.ru

Поступила в редакцию: 17.11.2025

Корректная автоматизация доступа к информации может быть выполнена за счет создания инструментальных средств, основанных на теории межмодельных отображений и обеспечении коммутативности преобразований данных. Данная статья посвящена технологии передачи данных между реляционной базой данных и табличным представлением данных специального вида. Структура таблицы является удобным средством работы пользователя, поскольку дает возможность не только редактировать данные, синхронизированные с базой данных, но и выполнять различные виды анализа с использованием электронных таблиц. В общем случае размер таблицы может быть огромным. В данной статье предлагается методика сокращения размера таблицы за счет логических ограничений при загрузке данных. При этом появляются две проблемы: фиктивные пустые значения и потеря пустых значений, необходимых для редактирования данных. В работе предложено решение этих проблем за счет использования промежуточного представления данных в виде запроса к базе данных, в котором присутствуют логические ограничения. Специальная форма этих ограничений, согласованная со стандартом SQL, является необходимым условием при решении проблемы пустых значений. Для этой цели формируются подмножества отношений из частичного порядка, который соответствует ссылочной целостности в базе данных. Полученные подграфы используются для формирования размерностей таблицы. В заключение статьи представлен анализ корректности преобразований.

Ключевые слова: реляционная модель данных, логические ограничения, коммутативность.

ОБРАЗЕЦ ЦИТИРОВАНИЯ

Зыкин С.В., Зыкин В.С., Шепелев Н.С. Синхронизация данных между таблицами специального вида и базой данных // Вестник ЮУрГУ. Серия: Вычислительная математика и информатика. 2025. Т. 14, № 4. С. 25–39. DOI: 10.14529/cmse250402.

Введение

Для взаимодействия с информационными ресурсами разрабатываются различные приложения, позволяющие делать выборочный просмотр и редактирование данных, представленных в удобном для пользователей формате. При этом существуют два подхода. Первый подход подразумевает, что приложение предоставляет пользователю регламентированный набор функций (например, просмотр расписания занятий на неделю в учебном заведении), что ограничивает возможности пользователей (например, при составлении семестрового расписания консультаций и отчетов) и является рутинной работой для программистов. Второй подход подразумевает гибкий механизм формирования пользовательских представлений данных на основе некоторых эвристических правил и предположений. Использование эвристик приводит к потере некоторой информации из поля зрения, либо к ошибочному появлению сведений и документов. Следовательно, на первый план выходит проблема корректности таких преобразований.

Проблему корректности интеграции неоднородных данных на основе межмодельных отображений впервые исследовал Л. А. Калиниченко. В работе [1] представлен способ формализации моделей данных, обеспечивающий корректное преобразование в процессе взаимодействия гетерогенных систем с несколькими базами данных (БД). В основе предложенной методики лежат коммутативные преобразования между разнородными моделями и центральной обобщенной моделью. Предложенная методика позволяет формировать корректные преобразования данных для одной модели, если произошли какие-либо изменения в другой. В данной статье методика используется для обоснования корректности преобразований.

В статье [2] рассматривается технология передачи данных из электронных таблиц в реляционную БД и обратно. При этом БД создается динамически с использованием методов анализа содержимого электронных таблиц. По содержимому таблиц определяются функциональные зависимости, строится их минимальное покрытие, что достаточно для построения отношений БД, удовлетворяющих требованиям третьей нормальной формы (3НФ). Загрузка БД осуществляется за счет применения аналога проекции к электронной таблице. В работе не обсуждается редактирования данных, а только массовая загрузка, и логические ограничения при этом не используются. Однако такой подход дает возможность сделать вывод о корректности преобразований.

В статье [3] рассматривается технология управления БД из электронных таблиц без использования макросов либо встроенных языков программирования. Для этой цели используются формулы в электронных таблицах (операторы реляционной алгебры реализуются с помощью функций). В качестве заготовки создается электронная таблица с пустыми рабочими листами для данных и рабочими листами, заполненными формулами для запросов. Когда пользователь вводит, изменяет или удаляет данные в рабочих листах данных, формулы в рабочих листах запросов автоматически вычисляют фактические результаты запросов к БД. Редактирование данных в БД с использованием электронных таблиц не предусмотрено. Логические ограничения на данные используются внутри функций. Корректность преобразований в сфере ответственности программиста.

Для автоматизации составления запросов при подготовке документов на предприятиях [4] была разработана математическая модель для генерации документов путем интеграции приложений Visual Basic for Application (VBA), ActiveX Data Object (ADO) и Extensible Markup Language (XML). На основе анализа типов данных документов разработаны файлы конфигурации, настроены шаблоны документов и использован алгоритм автоматической генерации документов. Данная модель была использована в практических задачах и показала свою эффективность. Технология не предусматривает редактирование данных и их возврат обратно в БД.

После того, как сформировано пользовательское представление данных возникает проблема их редактирования, и возврата новых значений в БД. Для этой цели необходимо воспользоваться методикой, подложенной в работе [1].

Проблема редактирования данных в БД с использованием пользовательских представлений данных исследовалась в работе [5]. Для обеспечения коммутативности преобразований использовалась промежуточная модель «Таблица связанных соединений», в которой дополнительный «вектор вхождений» позволял однозначно идентифицировать отношение и кортеж в БД, к которым относятся сделанные модификации. Основная идея формирования «Таблицы связанных соединений» заключается в переборе различных комбинаций

отношений БД и формировании промежуточной таблицы, если совокупность отношений в комбинации удовлетворяет условию соединения без потери информации (СБПИ). Далее промежуточные таблицы объединяются в единую таблицу с удалением подчиненных кортежей. В итоге получается NP-трудная задача. Следовательно, значительные интервалы времени формирования и преобразования «Таблицы связанных соединений» не позволяют выполнять работу в оперативном режиме со значительным объемом данных. В данной работе промежуточное представление данных состоит из одного запроса, что делает задачу полиномиальной, но усложняет задачу определения пустых значений (в «Таблице связанных соединений» эта задача решается сразу с использованием вектора вхождений). Кроме того, в данной статье анализ свойства СБПИ (квадратичный алгоритм по памяти и по времени) заменен проверкой выводимости зависимости с использованием линейного по памяти и по времени алгоритма.

Редактирование многотабличного запроса к БД рассмотрено в работе [6]. Поскольку результирующая таблица (запрос) является реляционной, то алгоритмы формирования этой таблицы и возврата отредактированных значений в БД удалось сформулировать в терминах реляционной алгебры без использования промежуточных представлений данных. Реализация технологии на PostgreSQL показала удовлетворительные результаты на тестовых БД. Недостатком является то, что реляционная таблица (запрос) не является удобным средством для работы пользователя.

В данной работе получила развитие технология межмодельных коммутативных преобразований [1]. Отказ от биективности состояний моделей данных [7] позволяет работать только с частью данных, накладывать логические ограничения на формируемые таблицы, что существенно улучшает восприятие их пользователем.

Статья оформлена следующим образом. В разделе 1 представлено описание задачи и обсуждение возникающих проблем. В разделе 2 рассмотрена структура и состав логических ограничений на данные при их загрузке из БД. Раздел 3 содержит описание способа формирования размерностей табличного представления данных и анализ размерностей. В разделе 4 рассмотрены алгоритмы формирования таблицы, синхронизация ее содержимого с БД и определены условия коммутативности преобразований. Обоснование корректности преобразований представлено в разделе 5. В заключении приводится краткая сводка результатов, полученных в работе, и указаны направления дальнейших исследований.

1. Обсуждение проблемы

В работе [8] рассматривается модель данных «Трансформация», частный случай которой впервые был представлен в работе [9] под названием «Семантическая трансформация». Для формального определения «Трансформации» рассмотрим следующие обозначения: $\mathcal{R} = \{R_1, R_2, \ldots, R_k\}$ — реляционная БД, где R_i — отношения (таблицы), удовлетворяющие требованиям 3НФ; $[R_i]$ — схема отношения R_i (заголовок таблицы); $R_i[S]$ — проекция R_i, по атрибутам S (вырезка по столбцам). U — конечное множество атрибутов, на которых задана БД. Связи между отношениями БД устанавливают ограничения ссылочной целостности и удовлетворяют типизированным ациклическим зависимостям включения [10].

Определим трансформацию Tr как табличное представление данных, где значения множества атрибутов $X = \{X_1, X_2, ..., X_m\}$ задают наименования строк, а значения множества атрибутов $Y = \{Y_1, Y_2, ..., Y_n\}$ задают наименования столбцов. Непустые множества атрибутов $X \subset U$ и $Y \subset U$ далее будем называть размерностями.

Особенность трансформации в том, что первые m столбцов таблицы будут озаглавлены именами атрибутов X_i. Последующие столбцы будут иметь составные имена из n значений y_{j,l_j} атрибутов Y_j, $j = \overline{1,n}$. Содержимое i-й строки таблицы состоит из значений:

$$(x_{i,1}, x_{i,2}, \ldots, x_{i,m}, z_{i,(m+1)}, z_{i,(m+2)}, \ldots, z_{i,(m+p)}),$$

где p — количество столбцов в таблице, озаглавленных значениями атрибутов Y. В i-й строке и j-м столбце, $j = \overline{1,m}$, содержится значение атрибута X_j. В последующих столбцах i-й строки содержатся значения $z_{i,l}$ атрибута $Z \in U$, который далее будем называть мерой. В рассматриваемой технологии значения меры подлежат редактированию с передачей новых значений в БД.

С практической точки зрения не целесообразно наличие общих атрибутов в размерностях. Кроме того, мера не должна принадлежать размерностям. Пусть $X \cap Y = \emptyset$, $Z \notin X \cup Y$. Структура трансформации представлена в табл. 1.

Таблица 1. Пример трансформации

X_1	X_2	X_3	y_{11}				y_{12}			
			y_{21}		y_{22}		y_{21}		y_{22}	
			y_{31}	y_{32}	y_{31}	y_{32}	y_{31}	y_{32}	y_{31}	y_{32}
x_{11}	x_{21}	x_{31}	z_{11}		z_{13}	z_{14}		z_{16}		z_{18}
		x_{32}					z_{25}			
	x_{22}	x_{31}		z_{32}		z_{34}			z_{37}	
		x_{32}	z_{41}		z_{43}			z_{46}		
x_{12}	x_{21}	x_{31}	z_{51}			z_{54}			z_{57}	
		x_{32}		z_{62}			z_{65}			z_{68}
	x_{22}	x_{31}	z_{71}		z_{73}			z_{76}		
		x_{32}	z_{81}			z_{84}			z_{87}	

В табл. 1 x_{ip} — значения атрибута X_i, y_{js} — значения атрибута Y_j, z_{ij} — значения атрибута Z.

Для каждой размерности строится подграф из частичного порядка, содержащий атрибуты этой размерности. При этом, множества атрибутов X и Y упорядочиваются в таблице Tr так, чтобы на верхних уровнях находились атрибуты с меньшим количеством значений, что сокращает количество дублирований на нижних уровнях. В работе [11] предложен алгоритм для построения таких иерархий в размерностях, в котором используются функциональные и многозначные зависимости исходной реляционной БД и мощности соответствующих доменов. Для удобства восприятия таблицы значения атрибутов X и Y упорядочиваются в каком-либо лексикографическом порядке.

Определение 1. Рассмотрим произвольное отношение $R_i \in \mathcal{R}$, пусть W и S – некоторые подмножества атрибутов $[R_i]$. Будем говорить, что в R_i реализована *функциональная зависимость* $W \to S$, если для любой реализации R_i не могут присутствовать два кортежа $t_1, t_2 \in R_i$, такие что $t_1[W] = t_2[W]$ и $t_1[S] \neq t_2[S]$.

Определение 2. Множество атрибутов V отношения R_i является *потенциальным ключом*, если для любого кортежа $t \in R_i$ совокупность значений $t[V]$ может присутствовать только в кортеже t.

Замечание 1. Один из потенциальных ключей отношения R_i, не содержащий избыточных атрибутов, назначается первичным ключом R_i.

Очевидным условием существования таблицы Tr является невозможность присутствия в одной ячейке таблицы более одного значения атрибута меры Z. Это условие реализуется за счет использования функциональных зависимостей в исходной БД. Для множества отношений $R = \{R'_1, R'_2, \ldots, R'_p\} \subseteq \mathcal{R}$, участвующих в формировании Tr, выделяются соответствующие им индексные файлы, обладающие свойством уникальности. По выделенным индексным файлам определяется множество реализованных (выполнимых) в БД функциональных зависимостей $F = \{F_1, F_2, \ldots, F_q\}$. Заметим, что зависимость $W \to S$ будет выводима, если атрибуты S принадлежит замыканию W_F^+ на множестве зависимостей F [12, 13]. В силу полноты и непротиворечивости системы аксиом функциональных зависимостей [12] выводимая зависимость является выполнимой в БД, а выполнимая зависимость выводимой. Следовательно, далее достаточно проверять только выводимость зависимостей.

В качестве промежуточного представления данных будем использовать запрос «проекция–селекция–соединение» (1), представленный в терминах реляционной алгебры. Для управления размером таблицы Tr будем использовать логические ограничения на значения атрибутов.

$$Q = \pi_{XYZ}(\sigma_L(R'_1 \bowtie R'_2 \bowtie \ldots \bowtie R'_p)), \tag{1}$$

где $\bowtie$ — операция естественного соединения; π — операция проекции, σ — операция селекции, L — логическая формула на атрибутах отношений R'_i. Кортежи в соединении отношений:

$$Q_1 = R'_1 \bowtie R'_2 \bowtie \ldots \bowtie R'_p,$$

подстановка которых в формулу L дает значение FALSE или UNKNOWN, будут отсутствовать в Q. Заметим, что Q_1 является универсальной реляционной таблицей, удовлетворяющей требованиям первой нормальной формы (1НФ). Далее будем считать, что Q_1 соответствует фактическому содержанию БД на отношениях R. Результат выполнения запроса Q является источником для формирования трансформации Tr.

Отсутствие значения атрибута Z в ячейке таблицы Tr означает, что в R значениям атрибутов X и Y не сопоставлено ни одного значения атрибута Z, то есть, в Q_1 отсутствует соответствующий кортеж. Ситуация меняется при наложении ограничения L. Логическое ограничение может удалить соответствующий кортеж в Q_1. В этом случае в R векторам значений $\vec{x_i} = (x_{i,1}, x_{i,2}, \ldots, x_{i,m})$ и $\vec{y_j} = (y_{1,j}, y_{2,j}, \ldots, y_{n,j})$ соответствует значение z_{ij}, а в таблице Tr соответствующая ячейка оказывается пустой. Такая ситуация будет вводить в заблуждение пользователя, и он может попытаться заполнить эту ячейку новым значением $\hat{z}_{ij}$, что является ошибкой. Возможны ситуации, когда z_{ij} в таблице Tr не пусто, а в БД этому значению не сопоставлены вектора $\vec{x_i} = (x_{i,1}, x_{i,2}, \ldots, x_{i,m})$ и $\vec{y_j} = (y_{1,j}, y_{2,j}, \ldots, y_{n,j})$, или этим векторам в БД сопоставлено другое значение $\hat{z}_{ij}$.

Самый простой вариант ошибочного содержимого формулы L — использование ограничения на значения меры Z. В общем случае причина появления ошибочно пустого значения в таблице Tr определена в утверждении 1.

Утверждение 1. Пусть кортеж t принадлежит Q_1, а в Q он удален ($L(t)$=FALSE), но остался кортеж t_1 совпадающий с t по Y ($L(t_1)$=TRUE), но отличающийся по X, и остался кортеж t_2 совпадающий с t по X, но отличающийся по Y ($L(t_2)$=TRUE). Тогда в Tr на

пересечении строки $\vec{x_i}$ и столбца $\vec{y_j}$ будет находиться фиктивное пустое значение, тогда как в БД ему соответствует значение $t[Z]$.

Причиной появления значения z_{ij}, отличного от фактического в БД, может быть разрушение зависимости $XY \to Z$.

2. Структура и состав логических ограничений

Формулу L будем задавать в виде конъюнкции логических условий, определенных на различных атрибутах множества U:

$$L = L_1 \wedge L_2 \wedge \ldots \wedge L_s. \tag{2}$$

В формуле (2) L_i, $i = \overline{1,s}$, дизъюнкция атомарных логических выражений:

$$L_i = l_{i1}(A_i) \vee l_{i2}(A_i) \vee \ldots l_{iv}(A_i), \tag{3}$$

где логические выражения l_{ij}, $j = \overline{1,v}$, определены для **одного** атрибута $A_i \in X \cup Y$.

С целью последующего создания программного обеспечения (ПО) атомарные выражения l_{ij} должны быть согласованы со стандартом языка SQL:

1. Арифметические операции сравнения $A_i \Theta\, const$, где операция Θ принимает одно из значений классической шестерки ($\Theta \in \{=, \neq, >, <, \geq, \leq\}$), константа $const$, должна быть согласована по типу с атрибутом A_i (Θ-сравнимы), в SQL сравнимыми являются числа в различных форматах, даты, время ...
2. Операция A_i BETWEEN $const_1$ AND $const_2$ — значение атрибута A_i должно удовлетворять интервальному ограничению. $const_1 \leq A_i \leq const_2$. Операция A_i NOT BETWEEN $const_1$ AND $const_2$ требует, чтобы значение атрибута A_i находилось за пределами указанного интервала.
3. Операция A_i IN $List$, где List список значений, получит значение TRUE, если в текущем кортеже значение атрибута A_i будет совпадать с каким-либо значением в списке $List$. Операция A_i NOT IN $List$ получит значение TRUE, если значение атрибута A_i будет отсутствовать в списке $List$.
4. Ограничение на символьные строки задаются в операции A_i LIKE Str. Операция получит значение TRUE, если строка, значение атрибута A_i, содержит в себе строку Str, заданную шаблоном. Обратная операция имеет вид A_i NOT LIKE Str.

Перечисленные варианты операций используют только часть операций языка SQL [14]. Например, не используется предикат EXISTS, поскольку в нем не заданы имена атрибутов, что исключает возможность явного управления размерностями трансформации Tr.

Естественным требованием к размерностям X и Y является отсутствие неопределенных значений. Для этого достаточно расширить логическую формулу L в запросе (1). С учетом структуры логической формулы (2) в нее дополняются условия определенности: конъюнкция операторов $A_i \neq emp$, где emp обозначает пустое значение, (в команде SQL: A_i IS NOT NULL) для атрибутов A_i, которые не входят в логическое выражение L, но принадлежат одной из размерностей. Если атрибут A_i входит в L, то на неопределенном значении атрибута формула примет значение UNKNOWN, что по умолчанию приведет к удалению соответствующего кортежа в Q_1 вместе с неопределенным значением.

Далее необходимо определиться со способом формирования размерностей Tr и их взаимодействием с формулой L.

3. Анализ размерностей в трансформации Tr

Рассмотрим фрагмент схемы БД учебного заведения (см. рис. 1).

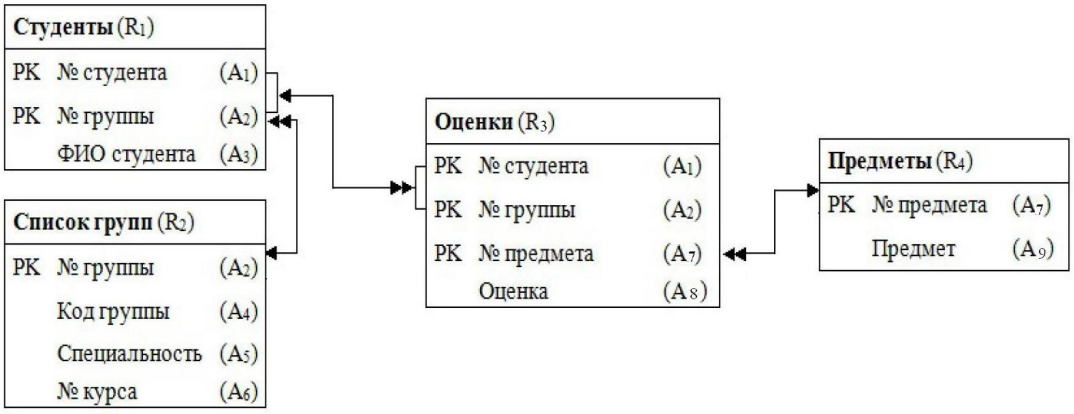

Рис. 1. Фрагмент схемы БД

Обозначение PK соответствует компонентам первичного ключа, стрелками показана ссылочная целостность, где сдвоенные стрелки указывают на внешние ключи. В скобках на схеме приведены символические имена атрибутов и отношений.

Самый простой способ формирования размерностей X и Y — получение проекций по этим множествам атрибутов: $\pi_X(Q)$ и $\pi_Y(Q)$, однако, при этом возможны потери некоторых строк и столбцов в таблице Tr даже если для них логическая формула не принимает значение FALSE или UNKNOWN. Для демонстрации этого результата сформируем приложение «Сводная ведомость» (см. табл. 2), где $X = \{R_2.A_2,\ R_2.A_4,\ R_1.A_3\}$, $Y = \{R_4.A_7,\ R_4.A_9\}$, $Z = \{R_3.A_8\}$. Если вычислить X и Y с использованием проекций $\pi_X(Q)$ и $\pi_Y(Q)$,

Таблица 2. Приложение «Сводная ведомость»

№ группы	Код группы	№ студента	ФИО студента	1 Физика	2 Химия	3 История	4 Биология	... ...
1	Б-211	1	Иванов	3		5		
		2	Петров		4			
		3	Сидоров	4			3	
2	Х-212	1	Ковалев	3		4		
		2	Попов		4		5	

то строки для групп Б-211 и Х-212 будут отсутствовать в табл. 2, если экзамены по всем предметам не проводились. Либо будут отсутствовать столбцы, если экзамены по соответствующим предметам не проводились. Такой результат неприемлем, указанные строки и столбцы должны присутствовать в таблице с пустыми ячейками для проставления оценок. В этом случае формирование размерности X выполняется по формуле:

$$Dim(X) = \pi_X(\sigma_{L_X}(R_1 \bowtie R_2)),$$

где L_X логическое ограничение на атрибут «Код группы» и/или на атрибут «ФИО студента». Формирование размерности Y выполняется по формуле:

$$Dim(Y) = \pi_Y(\sigma_{L_Y}(R_4)),$$

где L_Y логическое ограничение на атрибут «Предмет». Способ формирования L_X и L_Y из формулы L обсуждается далее.

Определение 3. Для обеспечения корректности преобразования данных в БД определим следующие три ограничения:

1. Отношения $R = (R'_1,\ R'_2,\ldots,\ R'_p)$ образуют частично упорядоченное множество относительно ссылочной целостности: $R'_i \preccurlyeq R'_j$ если R'_i является главным отношением, а R'_j — внешним. Частичный порядок содержит один максимальный элемент $R'_Z \in R$ ($Z \in [R'_Z]$), и изолированный элемент принадлежит частичному порядку, если $p = 1$. Отношение $R'_Z \in R$ соответствует семантике приложения, поскольку в нем будут реализованы операции редактирования БД.
2. В отношении R_Z множество атрибутов V является потенциальным ключом, для которого $V \subseteq XY$, $Z \notin V$ и $V \to XY$.
3. Только одно отношение R'_Z в R содержит атрибут Z.

Замечание 2. В рассмотренном приложении $V = \{R_3.A_1,\ R_3.A_2,\ R_3.A_7\}$, $R_4 \preccurlyeq R_3$, $R_2 \preccurlyeq R_1 \preccurlyeq R_3$, в силу транзитивности $R_2 \preccurlyeq R_3$.

Частичный порядок на отношениях БД может иметь вид, представленный на рис. 2. Связи задают ссылочные ограничения целостности, соответствующие типизированным за-

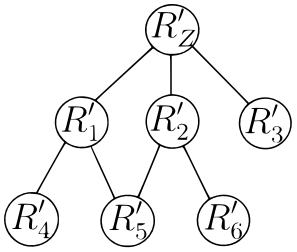

Рис. 2. Частичный порядок для ограничений целостности

висимостям включения [10]. Максимальный элемент частичного порядка R_Z на рис. 2 является внешним отношением для всех остальных отношений R_1 ... R_6 (все они являются главными для R_Z). R_1 является внешним для отношений R_4 и R_5, R_2 является внешним для отношений R_5 и R_6. Частичный порядок позволит вводить во внешние отношения только те данные, которые имеют совпадающие значения связанных атрибутов в главных отношениях. В противном случае ввод данных будет заблокирован системой управления базой данных (СУБД).

4. Формирование содержимого трансформации Tr

Ранее было отмечено, что в трансформации Tr не должно быть фиктивных пустых ячеек для атрибута Z с одной стороны. С другой стороны должны присутствовать реальные пустые ячейки для отсутствующих в БД значений атрибута Z. Для решения этой задачи предлагается правило формирования размерностей $Rule_1$. Рассмотрим это правило для размерности X.

1. Формируется множество отношений R_X из множества отношений R. Каждое из отношений R_X должно содержать атрибут множества X.
2. Поиск ближайших общих предков в частичном порядке для множества отношений R_X (см. замечание 3). Выбор одного из предков для построения размерности по X.

3. Формируется подграф из всех потомков выбранного предка. Все вместе они образуют множество отношений $\{R_{X_1}, R_{X_2}, \ldots, R_{X_d}\}$. В этом случае формирование размерности X выполняется по формуле:

$$Dim(X) = \pi_X(\sigma_{L_X}(R_{X_1} \bowtie R_{X_2} \bowtie \ldots \bowtie R_{X_d})),$$

где L_X логическое ограничение, сформированное по правилу $Rule_2$ (далее).

Замечание 3. Для поиска ближайших общих предков в частичном порядке целесообразно воспользоваться универсальным алгоритмом топологической сортировки Кана [15], либо какой-либо модификацией алгоритма Тарьяна [16], который более эффективен при поиске, но требует предварительной обработки графа. Если направленный граф является деревом, то общий ближайший предок один, но в частичном порядке их может оказаться несколько. Множества потомков для каждого из предков будут отличаться друг от друга не только по составу, но и по семантике. Следовательно, выбор общего предка должен сделать пользователь из предложенного списка, полученного в алгоритме.

При формировании размерностей используются логические ограничения, сформированные по правилу $Rule_2$. Рассмотрим это правило для атрибутов размерности X.

- В формуле (2) конъюнктивный компонент L_i, определенный для атрибута A_i, заменяем значением TRUE если $A_i \in Y \setminus X$, в противном случае компонент оставляем без изменения. Полученную формулу обозначим L_X.

Замечание 4. Применив правило $Rule_1$ к множеству атрибутов Y получим размерность

$$Dim(Y) = \pi_Y(\sigma_{L_Y}(R_{Y_1} \bowtie R_{Y_2} \bowtie \ldots \bowtie R_{Y_g})),$$

где логическая формула L_Y сформирована по правилу $Rule_2$, в котором атрибуты X и Y меняются местами, g — количество отношений в подграфе, выделенном в частичном порядке.

Рассмотрим схему алгоритма Alg_1 для формирования Tr:

- Шаг 1. Упорядочивается множество атрибутов X и отдельно упорядочивается множество атрибутов Y для построения оптимальной иерархии размерностей в соответствии с алгоритмом [11].
- Шаг 2. Заголовками строк в Tr становится таблица $Dim(X)$, атрибуты в которой упорядочены в соответствии с шагом 1, а строки отсортированы по значениям атрибутов с приоритетом, соответствующим порядковому номеру атрибута в X, полученному на предыдущем шаге. Заголовками первых m столбцов становятся атрибуты X. Остальные столбцы озаглавлены транспонированной таблицей $Dim(Y)$, прошедшей преобразования, аналогичные $Dim(X)$.
- Шаг 3. Последовательно просматриваются кортежи Q. Очередное значение атрибута Z помещаются в ячейку Tr, соответствующую заголовку строки ($x_{i,1}$, $x_{i,2}$, …, $x_{i,m}$) и заголовку столбца (y_{1,l_1}, y_{2,l_2}, …, y_{n,l_n}), значения которых получены из текущего кортежа Q.

Определим операцию по преобразованию атрибута Z в таблице Tr.

Определение 4. Единичной операцией Op будем считать замену одного значения z_{ij} на новое значение $\hat{z}_{ij}$ атрибута Z в таблице Tr. Значения z_{ij} и $\hat{z}_{ij}$ могут быть пустыми, либо непустыми. Для дальнейших операций сохраняется старое значение z_{ij}.

Алгоритм Alg_2 содержимое БД преобразует в соответствии с операцией Op. Рассмотрим схему этого алгоритма:

- Шаг 1. В таблице Tr определяются векторы для строки $\vec{x_i} = (x_{i,1}, x_{i,2}, \ldots, x_{i,m})$ и для столбца $\vec{y_j} = (y_{1,j}, y_{2,j}, \ldots, y_{n,j})$, соответствующие измененному значению $\hat{z}_{ij}$.
- Шаг 2. Поиск кортежа $t \in Q$, для которого $t[XY] = (\vec{x_i}, \vec{y_j})$. Если кортеж найден, то переход на шаг 3, в противном случае переход на шаг 4.
- Шаг 3. В отношении R_Z поиск кортежа u, для которого $u[V] = t[V]$. Выполняется замена $u[Z] = \hat{z}_{ij}$. Конец алгоритма.
- Шаг 4. Пусть $S = [R_Z] \cap (X \cup Y)$ В отношение R_Z дополняется новый кортеж u: $u[S \cap X] = \vec{x_i}[S \cap X]$, $u[S \cap Y] = \vec{y_j}[S \cap Y]$ и $u[Z] = \hat{z}_{ij}$. Атрибуты, не вошедшие в множество $S \cup Z$, получают значение emp. Если СУБД прервала операцию в связи с нарушением ограничений целостности в БД, то переход на шаг 5, иначе конец алгоритма.
- Шаг 5. Информирование пользователя о возникшей ошибке, возврат значения z_{ij} на прежнее место в Tr. Конец алгоритма.

Замечание 5. Поскольку кортеж u является частью кортежа t, $V \subseteq [R_Z]$ и $V \subseteq XY$, то на шаге 3 u существует. Благодаря транзитивной зависимости $V \to Z$ и 3НФ кортеж u единственный в R_Z.

Замечание 6. Алгоритм Alg_2 целесообразно выполнять асинхронно с редактированием Tr сразу после операции Op.

После определения всех необходимых компонентов коммутативность преобразований с использованием (1), Alg_1, Alg_2 и Op можно представить в следующем виде:

$$DB' \xrightarrow{(1)} Q' \xrightarrow{Alg_1} Tr' \xrightarrow{Op} Tr'', \tag{4}$$

$$DB' \xrightarrow{Alg_2} DB'' \xrightarrow{(1)} Q'' \xrightarrow{Alg_1} Tr'', \tag{5}$$

где DB' и DB'' — начальное и конечное состояния БД, Tr' и Tr'' — начальное и конечное состояния трансформации, Q' и Q'' — результаты выполнения запроса Q до и после операции Op. Следовательно, из DB' в Tr'' можно перейти двумя различными путями, но результат должен быть один и тот же, что гарантирует корректность выполненных преобразований в БД.

5. Анализ корректности преобразований

Не сложно убедиться в истинности следующего утверждения, в котором устанавливается связь между трансформацией и функциональными зависимостями.

Утверждение 2. В таблице Tr отсутствует наложение различных значений атрибута Z в одной ячейке тогда и только тогда, когда Q удовлетворяет функциональной зависимости $XY \to Z$.

Действительно, достаточность условия следует из определения 1, необходимость следует из условия существования Tr.

Следующая лемма дает достаточное условие для реализации зависимости в промежуточном представлении данных Q, следовательно, и в Tr.

Лемма 1. Q удовлетворяет функциональной зависимости $XY \to Z$, если она реализована в БД на выделенном множестве отношений R.

Доказательство. Пусть F реализованное множество зависимостей на R. Рассмотрим произвольную зависимость $F_i \in F$. По построению F существует отношение R'_j, в котором зависимость F_i реализована. Произвольный кортеж t в R'_j становится частью кортежа в Q_1 в неизменном виде, поскольку операция естественного соединения выполняется по совпадению значений одноименных атрибутов в кортежах других отношений. Следовательно, все кортежи в Q_1 удовлетворяют зависимостям F. По условию леммы зависимость $XY \to Z$ реализована в R (выводима из F), следовательно, она реализована в Q_1. Поскольку логическая формула L и последующая проекция только удаляют кортежи из Q_1, то они не могут нарушить зависимость $XY \to Z$ на множестве кортежей Q. □

Замечание 7. Утверждение 2 и лемма 1 гарантируют, что в таблице Tr не появятся непустое значение z_{ij}, отличное от фактического значения в БД.

Теорема 1. Результаты преобразований (4) и (5) совпадают и не содержат фиктивных пустых значений.

Доказательство. После выполнения преобразований (4) будут определены векторы $\vec{x_i}$, $\vec{y_i}$ и значения атрибута Z: старое z_{ij} и новое $\hat{z}_{ij}$. Доказательство построим на сравнении компонентов преобразований (4) и (5).

Пусть алгоритм Alg_2 завершился выполнением шага 3. Состояния DB' и DB'' различаются одним значением атрибута Z в единственном ($V \subseteq XY$) кортеже u отношения R_Z: $u[Z] = z_{ij}$ и $u[Z] = \hat{z}_{ij}$ соответственно. Множество зависимостей F при этом не изменится. Поскольку атрибут Z принадлежит только R_Z, то результаты выполнения запросов Q' и Q'' будут отличаться значением этого атрибута в кортежах $t[XY] = (\vec{x_i}, \vec{y_i})$. Остальные значения в кортежах Q' и Q'' будут совпадать. Поскольку размерности $Dim(X)$ и $Dim(Y)$ остаются без изменений, то в результате преобразований (5) будет получено представление данных Tr''.

Пусть Alg_2 успешно завершил работу на шаге 4. При дополнении нового кортежа $u \in R_Z$ множество зависимостей F не изменится, поскольку $V \subseteq XY$ и V является потенциальным ключом в R_Z. За счет нового кортежа u в Q'' появится множество новых кортежей T, которых не было в Q'. Среди кортежей T найдется хотя бы один кортеж t с определенными значениями XY, поскольку соответствующие размерности есть в Tr'. Остальные будут отсеяны расширенной логической формулой L. Для кортежа t выполнено $t[V] = u[V]$ по свойству операции естественного соединения, следовательно, $t[XY] = (\vec{x_i}, \vec{y_i})$, поскольку V потенциальный ключ и зависимость $V \to XY$ выводима из F. Из леммы 1 следует, что $t[Z] = u[Z]$. Следовательно, в результате преобразований (5) получим трасформацию Tr''.

Причина появления фиктивного пустого значения определена в условии 1. Способ построения размерностей на основе подграфа и структура логического ограничения L (конъюнкция ограничений) гарантируют, что в Tr не появятся соответствующие строки и столбцы, следовательно, не появится фиктивное пустое значение. □

Замечание 8. Раздельное формирование размерностей не препятствует появлению всех допустимых строк и столбцов, а использование только иерархий в размерностях позволяет увеличить количество пустых ячеек, доступных для редактирования. В конкретных ситуациях можно подобрать ограничения и размерности, которые дают большее количество не фиктивных пустых значений, например, использовать дизъюнкцию для различных атрибутов. Однако в общем случае это может привести к ошибкам. К фиктивным пустым ячейкам в Tr приводит использование в логической формуле L не принадлежащих размерностям атрибутов, в том числе, когда они находятся в отношениях без атрибутов размерностей. Тогда выполнение условия 3 будет недостаточно, потребуются дополнительные ограничения в виде зависимостей функциональных и/или соединения.

Заключение

В данной статье получила развитие технология автоматического формирования пользовательского представления данных из реляционной БД с использованием таблицы «Трансформация». Логические ограничения при формировании «Трансформации» позволяют сделать ее обозримой при работе с произвольными объемами данных. Основной проблемой является возможное присутствие в таблице фиктивных пустых значений и возможное отсутствие реальных пустых значений. Для решения этой проблемы использован специальный вид логических ограничений, и способ формирования размерностей таблицы, основанный на ссылочной целостности в БД.

Структура и содержимое «Трансформации» позволяют использовать ее для анализа данных. В статье предложены алгоритмы синхронизации измененных данных в «Трансформации» с БД. Корректность достигается за счет коммутативности соответствующих преобразований.

В работе [8] представлено экспериментальное программное обеспечение, разработанное в среде Microsoft Office, которое позволило выявить проблемные места в рассматриваемой технологии и найти для них решение. В ближайших планах находится продолжение исследований «Трансформации» с целью ослабления ограничений на ее структуру и состав: увеличение количества одновременно редактируемых атрибутов БД, использование при редактировании внешних таблиц БД. Разработка экспериментального программного обеспечения планируется в среде PostgreSQL и Microsoft Office.

Работа выполнена в рамках государственного задания ИМ СО РАН, проект FWNF-2022-0016.

Литература

1. Kalinichenko L.A. Methods and tools for equivalent data model mapping construction // Advances in Database Technology - EDBT'90. Vol. 416. Springer, 1990. P. 99–119. Lecture Notes in Computer Science. DOI: 10.1007/BFb0022166.

2. Cunha J., Saraiva J., Visser J. From spreadsheets to relational databases and back // Proceedings of the 2009 ACM SIGPLAN Workshop on Partial Evaluation and Program Manipulation, Savannah, GA, USA, 2009. ACM, 2009. P. 79–188. DOI: 10.1145/1480945.1480972.

3. Tyszkiewicz J. Spreadsheet as a relational database engine // Proceedings of the 2010 ACM SIGMOD International Conference on Management of data (SIGMOD '10). ACM, New York,

NY, USA, 2010. P. 195–206. DOI: 10.1145/1807167.1807191.

4. Mi L., Li C., Du P., *et al.* Construction and application of an automatic document generation model // 26th International Conference on Geoinformatics, Kunming, China, 2018. P. 1–6. DOI: 10.1109/GEOINFORMATICS.2018.8557127.
5. Редреев П.Г. Построение табличных приложений со списочными компонентами // Информационные технологии. 2009. № 5. С. 7–12.
6. Зыкин В.С., Цымблер М.Л. Обновление многотабличных представлений на основе коммутативных преобразований базы данных // Вестник ЮУрГУ. Серия: Вычислительная математика и информатика. 2019. Т. 8, № 2. С. 92–106. DOI: 10.14529/cmse190206.
7. Zykin S.V., Zykin V.S. Commutative Transformations in Multi-Model Databases // Dynamics of Systems, Mechanisms and Machines (Dynamics), Omsk, Russian Federation, 2023. P. 1–4. DOI: 10.1109/Dynamics60586.2023.10349549.
8. Zykin S.V., Poluyanov A.N., Zykin V.S. Tool Environment for Editing Spreadsheet Applications // Dynamics of Systems, Mechanisms and Machines (Dynamics), Omsk, Russian Federation, 2024. P. 1–5. DOI: 10.1109/Dynamics64718.2024.10838693.
9. Цаленко М.Ш. Моделирование семантики в базах данных. М.: Наука, 1989. 288 с.
10. Koehler H., Link S. Inclusion dependencies and their interaction with functional dependencies in SQL // J. Comput. Syst. Sci. 2017. Vol. 85. P. 104–131. DOI: 10.1016/j.jcss.2016.11.004.
11. Редреев П.Г. Построение иерархий в многомерных моделях данных // Известия Саратовского университета. Серия Математика. Механика. Информатика. 2009. Т. 9, № 4, ч. 1. С. 84–87. DOI: 10.18500/1816-9791-2009-9-4-1-84-87.
12. Ульман Дж. Основы систем баз данных. М.: Финансы и статистика, 1983. 334 с.
13. Мейер Д. Теория реляционных баз данных. М.: Мир, 1987. 608 с.
14. Мосин С.В. Сравнение областей истинности запросов к реляционной базе данных // Вестник ЮУрГУ. Серия: Вычислительная математика и информатика. 2016. Т. 5, № 1. С. 85–99. DOI: 10.14529/cmse160108.
15. Kahn A.B. Topological sorting of large networks // Communications of the ACM. 1962. Vol. 5, no. 11. P. 558–562. DOI: 10.1145/368996.369025.
16. Eswaran K.P., Tarjan R.E. Augmentation problems // SIAM J. on Computing. 1976. Vol. 5, no. 4. P. 653–665.

Зыкин Сергей Владимирович, д.т.н., профессор, ведущий научный сотрудник, заведующий лабораторией МППИ, Институт математики им. С.Л. Соболева СО РАН (Омск, Российская Федерация)

Зыкин Владимир Сергеевич, к.ф.-м.н., научный сотрудник, Институт математики им. С.Л. Соболева СО РАН (Омск, Российская Федерация)

Шепелев Никита Сергеевич, аспирант Омского государственного технического университета (Омск, Российская Федерация)

DOI: 10.14529/cmse250402

SYNCHRONIZATION OF DATA BETWEEN SPECIAL-TYPE TABLES AND THE DATABASE

© 2025 S.V. Zykin[1], V.S. Zykin[1], N.S. Shepelev[2]

[1] *Sobolev Institute of Mathematics SB RAS*
(ac. Koptyug avenue 4, Novosibirsk, 630090 Russia),
[2] *Omsk State Technical University (Mira avenue 11, Omsk, 644050 Russia)*
E-mail: zykin@ofim.oscsbras.ru, vszykin@mail.ru, n06k@mail.ru

Received: 17.11.2025

Correct automation of access to information can be achieved by creating tools based on the theory of inter-model mappings and ensuring the commutativity of data transformations. This paper is devoted to the technology of data transfer between a relational database and a special type of tabular representation of data. The table structure is a convenient tool for the user, since it allows not only to edit data synchronized with the database, but also to perform various types of analysis using spreadsheets. In general case, the table size can be huge. This paper proposes a technique for reducing table size by applying logical constraints when loading data. This introduces two problems: fictitious empty values and the loss of empty values needed to edit the data. The work proposes a solution to these problems by using an intermediate representation of data in the form of a database query that contains logical constraints. A special form of these constraints, consistent with the SQL standard, is necessary to deal with the null value problem. For this purpose, subsets of relations are formed from a partial order that corresponds to referential integrity in the database. The resulting hierarchies are used to form the table dimensions. The paper concludes with an analysis of the correctness of the transformations.

Keywords: relational data model, logical constraints, commutativity.

FOR CITATION

Zykin S.V., Zykin V.S., Shepelev N.S. Synchronization of Data between Special-Type Tables and the Database. Bulletin of the South Ural State University. Series: Computational Mathematics and Software Engineering. 2025. Vol. 14, no. 4. P. 25–39. (in Russian) DOI: 10.14529/cmse250402.

References

1. Kalinichenko L.A. Methods and tools for equivalent data model mapping construction. Advances in Database Technology - EDBT'90. Vol. 416. Springer, 1990. P. 99–119. Lecture Notes in Computer Science. DOI: 10.1007/BFb0022166.

2. Cunha J., Saraiva J., Visser J. From spreadsheets to relational databases and back. Proceedings of the 2009 ACM SIGPLAN Workshop on Partial Evaluation and Program Manipulation, Savannah, GA, USA, 2009. ACM, 2009. P. 79–188. DOI: 10.1145/1480945.1480972.

3. Tyszkiewicz J. Spreadsheet as a relational database engine. In Proceedings of the 2010 ACM SIGMOD International Conference on Management of data (SIGMOD '10). ACM, New York, NY, USA, 2010. P. 195–206. DOI: 10.1145/1807167.1807191.

4. Mi L., Li C., Du P., *et al.* Construction and application of an automatic document generation

model. 26th International Conference on Geoinformatics, Kunming, China, 2018. P. 1–6. DOI: 10.1109/GEOINFORMATICS.2018.8557127.

5. Redreev P.G. Construction of Applications with the List Components Information Technology. 2009. No. 5. P. 7–12.
6. Zykin V.S., Zymbler M.L. Updating Multi-table Views Based on Commutative Database Transformations. Bulletin of the South Ural State University. Computational Mathematics and Software Engineering. 2019. Vol. 8, no. 2. P. 92–106. (in Russian) DOI: 10.14529/cmse190206.
7. Zykin S.V., Zykin V.S. Commutative Transformations in Multi-Model Databases. Dynamics of Systems, Mechanisms and Machines (Dynamics), Omsk, Russian Federation, 2023. P. 1–4. DOI: 10.1109/Dynamics60586.2023.10349549.
8. Zykin S.V., Poluyanov A.N., Zykin V.S. Tool Environment for Editing Spreadsheet Applications. Dynamics of Systems, Mechanisms and Machines (Dynamics), Omsk, Russian Federation, 2024. P. 1–5. DOI: 10.1109/Dynamics64718.2024.10838693.
9. Tsalenko M.Sh. Modeling semantics in databases. Moscow: Nauka, 1989. 288 p. (in Russian).
10. Koehler H., Link S. Inclusion dependencies and their interaction with functional dependencies in SQL. J. Comput. Syst. Sci. 2017. Vol. 85. P. 104–131. DOI: 10.1016/j.jcss.2016.11.004.
11. Redreev P.G. Construction of hierarchies in multidimensional data models. Bulletin of the Saratov University. Series Mathematics. Mechanics. Computer Science. 2009. Vol. 9, no. 4(1). P. 84–87. DOI: 10.18500/1816-9791-2009-9-4-1-84-87.
12. Ullman J. Principles of Database Systems. Moscow: Finance and Statistics, 1983. 334 p. (in Russian).
13. Meyer D. The Theory of Relational Databases. Moscow: Mir, 1987. 608 p. (in Russian).
14. Mosin S.V. Truth Space Comparison of Relational Database Queries. Bulletin of the South Ural State University. Computational Mathematics and Software Engineering. 2016. Vol. 5, no. 1. P. 85–99. DOI: 10.14529/cmse160108.
15. Kahn A.B. Topological sorting of large networks. Communications of the ACM. 1962. Vol. 5, no. 11. P. 558–562. DOI: 10.1145/368996.369025.
16. Eswaran K.P., Tarjan R.E. Augmentation problems. SIAM J. on Computing. 1976. Vol. 5, no. 4. P. 653–665.

УДК 004.032.26 **DOI: 10.14529/cmse250403**

АНАЛИЗ ПРОИЗВОДИТЕЛЬНОСТИ ВЫВОДА МОДЕЛЕЙ ГЛУБОКОГО ОБУЧЕНИЯ НА ПЛАТЕ BANANA PI BPI-F3 НА ПРИМЕРЕ ЗАДАЧИ КЛАССИФИКАЦИИ ИЗОБРАЖЕНИЙ

© 2025 И.С. Мухин, В.Д. Кустикова

Нижегородский государственный университет им. Н.И. Лобачевского
(603022 Нижний Новгород, пр. Гагарина, 23)
E-mail: ismukhin03@gmail.com, valentina.kustikova@itmm.unn.ru

Поступила в редакцию: 18.08.2025

В работе выполняется анализ производительности вывода известных нейросетевых моделей ResNet-50 и MobileNetV2, обеспечивающих решение задачи классификации изображений, на плате Banana Pi BPI-F3, которая построена на базе архитектуры RISC-V. Вывод запускается средствами доступных фреймворков: PyTorch, TensorFlow Lite, Apache TVM и ExecuTorch. Предварительно модели конвертируются в формат каждого целевого фреймворка. Выполняется проверка корректности решения задачи с использованием полученных нейронных сетей. Демонстрируется, что показатели качества классификации изображений для этих моделей хорошо соотносятся с опубликованными значениями. Далее выполняется подбор оптимальных параметров запуска вывода для каждого фреймворка и модели. Сравнительный анализ производительности вывода показывает, что ExecuTorch (с XNNPACK-бэкендом) для обеих моделей демонстрирует лучшие результаты. Для модели ResNet-50 показатель количества кадров, обрабатываемых за секунду (Frames per Second, FPS), меняется от 2.649 до 3.339 fps при оптимальных параметрах запуска в зависимости от размера входного набора данных, обрабатываемого за один прямой проход по сети, для MobileNetV2 — от 11.26 до 29.96 fps. TensorFlow Lite уступает ExecuTorch в среднем в ∼2.1 раза. PyTorch и Apache TVM демонстрируют более низкие показатели производительности. Предположительно это связано с тем, что вывод в этих фреймворках не в полной мере оптимизирован для процессоров архитектуры RISC-V.

Ключевые слова: глубокое обучение, классификация изображений, производительность вывода, PyTorch, TensorFlow Lite, Apache TVM, ExecuTorch, Banana Pi BPI-F3, RISC-V.

ОБРАЗЕЦ ЦИТИРОВАНИЯ

Мухин И.С., Кустикова В.Д. Анализ производительности вывода моделей глубокого обучения на плате Banana Pi BPI-F3 на примере задачи классификации изображений // Вестник ЮУрГУ. Серия: Вычислительная математика и информатика. 2025. Т. 14, № 4. С. 40–60. DOI: 10.14529/cmse250403.

Введение

Модели и методы глубокого обучения являются эффективными инструментами для решения различных прикладных задач [1]. Применение глубокого обучения начинается с построения архитектуры нейронной сети. Далее сеть *обучается* на выделенном наборе данных, называемом *тренировочной выборкой*, и выполняется тестирование на данных, которые модель «не видела» в процессе обучения. При достижении приемлемых показателей качества модель внедряется в реальную программную систему, иначе выполняется настройка параметров обучения и/или модификация архитектуры сети. *Внедрение* предполагает многократное решение задачи на новых данных, называемое выводом. *Вывод* — это прямой проход по сети с целью получения и последующей обработки ее выхода.

Анализ производительности вывода нейронных сетей является важным этапом в процессе их внедрения в приложения, которые, как правило, должны функционировать в режи-

ме реального времени на маломощных устройствах. RISC-V [2] — активно развивающаяся архитектура, которая имеет большие перспективы широкого распространения в подобных устройствах. Проведение исследований, связанных с анализом производительности вывода на устройствах RISC-V, способствует дальнейшему развитию системного и прикладного программного обеспечения для таких устройств. Также оно имеет значение для планирования специализированных расширений набора команд, ориентированных на ускорение вывода нейронных сетей.

Цель настоящего исследования состоит в том, чтобы провести сравнительный анализ производительности вывода двух широко известных моделей ResNet-50 [3] и MobileNetV2 [4], обеспечивающих решение задачи классификации изображений, при запуске на плате Banana Pi BPI-F3, которая построена на базе архитектуры RISC-V, с использованием доступных фреймворков глубокого обучения. Исследования по аналогичной тематике проводились ранее для других архитектур [5–8]. Обзор литературы показывает, что для RISC-V подобные работы только начинают появляться [9–15]. При этом авторы используют разные тестовые модели, библиотеки для вывода нейронных сетей и устройства RISC-V для проведения экспериментов. Для платы Banana Pi BPI-F3 результаты бенчмаркинга вывода до настоящего момента не публиковались. Также в отличие от существующих работ в данном исследовании используется наиболее полный спектр фреймворков, доступных в настоящее время для вывода на устройствах RISC-V (PyTorch [16], TensorFlow Lite [17], Apache TVM [18], ExecuTorch [19]). Наряду с этим, реализация вывода и необходимой инфраструктуры разрабатывается в рамках открытой программной системы бенчмаркинга вывода Deep Learning Inference Benchmark (DLI) [20, 21], вследствие чего является общедоступной и расширяемой с точки зрения тестовых моделей, поддерживаемых фреймворков глубокого обучения и устройств, используемых для запуска.

Работа построена следующим образом. В разделе 1 дается обзор по тематике анализа производительности вывода нейронных сетей на процессорах архитектуры RISC-V, отмечаются отличия от существующих исследований. Раздел 2 содержит описание фреймворков глубокого обучения, доступных для запуска на устройствах RISC-V. В разделе 3 приводится формальная постановка задачи классификации изображений с большим числом категорий. Далее (раздел 4) описывается методика анализа производительности вывода глубоких моделей, которая предлагается в [5] и используется в экспериментальной части настоящей работы. В разделе 5 рассматривается архитектура программной системы DLI и основные изменения, которые внесены в рамках данного исследования. Раздел 6 посвящен анализу результатов экспериментов, полученных при запуске вывода моделей ResNet-50 и MobileNetV2 на плате Banana Pi BPI-F3 средствами доступных фреймворков. В заключении приводится краткая сводка результатов, формулируются выводы.

1. Обзор литературы

Внедрение нейронных сетей в реальные программные системы — сложный и важный этап жизненного цикла глубоких моделей. Он предполагает, что построена нейросетевая модель, которая решает поставленную задачу с высокими показателями качества. Далее необходимо проанализировать производительность ее вывода на конкретном устройстве, где будет выполняться многократный запуск. В зависимости от полученных результатов анализа может осуществляться оптимизация сети, либо изменение архитектуры с целью получения более «легковесной» и качественной модели. Один из типовых подходов к оп-

тимизации — *квантование весов* [22]. Квантование предусматривает понижение точности весов от формата FP32 к INT8 или UINT8 без существенной потери качества решения задачи. Технически реализуется с помощью встроенных функций библиотек глубокого обучения, либо специализированных фреймворков (например, NNCF [23]). При этом следует учитывать, что используемый для вывода инструмент должен поддерживать запуск квантованных моделей.

В настоящем разделе рассмотрим исследования, которые наиболее близки к тематике анализа производительности вывода на устройствах RISC-V. Авторы [9] описывают процедуру включения и оптимизации P-расширения RISC-V с целью выполнения квантованных нейронных сетей в тензорном компиляторе Apache TVM. В экспериментальной части приводятся результаты бенчмаркинга группы моделей MobileNet-v1 и Inception-v3 с весами в форматах FP32 и INT8 на симуляторе Spike с поддержкой P-расширения RISC-V. В [10] предлагается сравнение двух машин на базе RISC-V и Raspberry Pi для нейросетевого вывода на примере группы моделей MobileNet, и даются рекомендации по использованию этих машин. Это системы Sipeed Maixduino с ускорителем сверточных нейросетей и Raspberry Pi 4B в сочетании с USB-ускорителем Coral от Google. Авторы [11] оценивают производительность широкого спектра рабочих нагрузок машинного обучения на RISC-V с использованием архитектурного симулятора с открытым исходным кодом gem5. Работа направлена на бенчмаркинг вывода нейросетевых моделей, обеспечивающих решение различных прикладных задач: классификация изображений, детектирование объектов, семантическая сегментация изображений, оценка глубины сцены и других. Цель работы [12] — оценить производительность вывода языковых моделей BERT и GPT-2 на 64-ядерной архитектуре RISC-V SOPHON SG2042 с поддержкой векторных инструкций RVV v0.7.1. Проводится бенчмаркинг моделей с включением RVV-расширения и без него, при этом используется OpenBLAS и BLIS в качестве бэкендов BLAS для фреймворка PyTorch. В [13] анализируется производительность вывода кодировщика в составе модели трансформера на трех маломощных платформах с архитектурой RISC-V. Выполняется исследование вывода для двух репрезентативных представителей семейства моделей BERT, выявляются узкие места и возможности оптимизации на процессорах RISC-V: XuanTie C906, C908 и C910. Авторы [14] исследуют энергоэффективность и производительность квантованных нейросетевых моделей, развернутых на маломощных устройствах с различными аппаратными архитектурами, включая RISC-V, x86, ARM 64 и ARM 32. Рассматриваются два принципиально разных типа моделей: рекуррентные нейронные сети и большие языковые модели. При этом оценивается, как эти модели работают в практических сценариях. Измеряется точность, время выполнения и энергопотребление.

Данная работа является развитием [15], в которой анализируется производительность вывода сети DenseNet-121 на плате Lichee Pi 4A при запуске средствами инструментария OpenVINO, библиотеки TensorFlow Lite и компилятора машинного обучения Apache TVM. В отличие от [15] в данном исследовании OpenVINO не участвует в сравнении вследствие того, что до настоящего момента многие нейросетевые преобразования не оптимизированы для RISC-V. Наряду с TensorFlow Lite и Apache TVM здесь еще рассматриваются фреймворки PyTorch и ExecuTorch. В отличие от других работ, представленных в обзоре, в исследовании используется другой набор тестовых моделей (ResNet-50 и MobileNetV2). При этом вывод запускается на устройстве Banana Pi BPI-F3. Также разрабатываемое программное решение, обеспечивающее сбор результатов качества и производительности вывода нейросетей,

является открытым (лицензия Apache 2.0) и расширяемым, и может быть использовано для исследования производительности вывода других моделей, фреймворков глубокого обучения и устройств RISC-V.

2. Фреймворки глубокого обучения для вывода на устройствах RISC-V

Для сравнения производительности вывода используется несколько широко известных инструментов глубокого обучения, которые портированы и оптимизированы (частично или полностью) их разработчиками для запуска на процессорах архитектуры RISC-V.

1. PyTorch [16] — обширная экосистема с открытым исходным кодом для решения прикладных задач с использованием машинного обучения. Инструмент разработан на базе библиотеки Torch. В настоящее время PyTorch является стандартом де-факто для обучения и тестирования нейронных сетей вследствие удобства реализации и расширяемости возможностей. В процессе его разработки использован положительный опыт создания более ранних фреймворков глубокого обучения. Разработчики PyTorch предоставляют интерфейсы для языков C++ и Python, а также обертки для Java.
2. TensorFlow Lite (с сентября 2024 года LiteRT) [17] — библиотека для развертывания глубоких нейросетевых моделей на мобильных устройствах и микроконтроллерах. Имеются интерфейсы для C++, Python и некоторых других языков программирования.
3. Apache TVM [18] — активно развивающийся компилятор моделей машинного обучения с открытыми исходными кодами. Цель разработки состоит в предоставлении инженерам инструмента для оптимизации и последующего эффективного вывода нейросетевых моделей на разных устройствах. Обеспечивается широкий спектр программных интерфейсов, далее в работе используется Python API.
4. ExecuTorch [19] — относительно новый фреймворк для вывода глубоких нейросетевых моделей на мобильных и периферийных устройствах, а также на микроконтроллерах. Он является частью экосистемы PyTorch Edge и обеспечивает эффективное развертывание различных моделей в формате PyTorch на маломощных устройствах. Разработчики предоставляют программные интерфейсы для C++, Python и ряда других языков. При этом поддерживается значительное количество бэкендов, гарантирующих эффективное исполнение низкоуровневых операций, которые возникают в нейронных сетях, на конкретном аппаратном обеспечении.

Следует отметить, что существуют другие фреймворки, обеспечивающие запуск глубоких нейронных сетей на устройствах RISC-V (например, ncnn [24]), но их использование затруднительно вследствие ограниченных возможностей доступных конвертеров моделей.

3. Постановка задачи классификации изображений

Задача классификации изображений состоит в том, чтобы определить категорию, которой принадлежит изображение, из допустимого набора классов. При решении задачи на входе глубокой нейросетевой модели имеется изображение I, как правило, в формате RGB с разрешением $w \times h$, где w — ширина, h — высота изображения соответственно. Изображение I представляется в виде трехмерной матрицы интенсивностей с элементами I_{ijk} , где $i \in \{0, 1, \ldots, w-1\}$, $j \in \{0, 1, \ldots, h-1\}$, $k \in \{0, 1, 2\}$. Интенсивности принимают целые неотрицательные значения в диапазоне от 0 до 255, либо вещественные значения в диапазоне от 0 до 1, если они нормированы. Выход классификационной нейронной сети — это

вещественный вектор, каждый элемент которого содержит достоверность принадлежности изображения к одной из допустимых категорий. Размер вектора соответствует количеству этих категорий N. Таким образом, нейросетевая модель обеспечивает построение отображения $\phi : I \to \mathbb{R}^N$. Цель решения задачи состоит в том, чтобы построить нейронную сеть, которая для входного изображения формирует вектор достоверностей, где индекс максимального значения отвечает номеру искомого класса.

4. Методика анализа производительности вывода

Общая схема анализа производительности вывода состоит из нескольких этапов [5].

1. Обучение и/или конвертация исходной глубокой модели в форматы различных фреймворков, которые предполагается использовать для ее вывода на конечном устройстве.
2. Анализ и сравнение качества работы полученных моделей для проверки корректности предыдущего этапа.
3. Определение оптимальных параметров для запуска вывода.
4. Сжатие и оптимизация моделей.
5. Анализ и сравнение качества оптимизированных моделей.
6. Сравнение производительности вывода с использованием полученного набора моделей.

Сжатие и оптимизация нейронных сетей выходит за рамки данного исследования. Поэтому последовательность изложения результатов соответствует пунктам 1, 2, 3 и 6.

5. Программная реализация

Deep Learning Inference Benchmark (DLI) [20, 21] — программная система, разрабатываемая в ННГУ, позволяющая собирать показатели качества и производительности вывода глубоких моделей в автоматическом режиме. Система предоставляет программные интерфейсы для вывода на языках C++ и Python, поддерживает вывод с использованием значительного количества широко известных фреймворков глубокого обучения. DLI включает следующие основные компоненты (рис. 1).

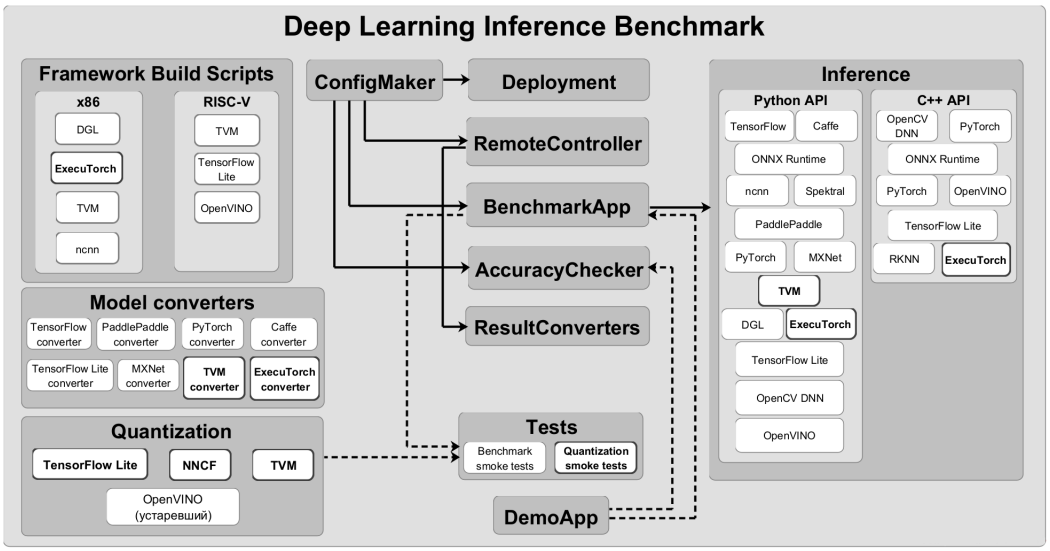

Рис. 1. Основные компоненты системы Deep Learning Inference Benchmark

1. **Framework Build Scripts** — независимый компонент, в состав которого входят скрипты для сборки различных фреймворков под архитектуры x86 и RISC-V.

2. **Model converters** — независимый компонент, содержащий программный интерфейс для конвертации моделей из формата одного фреймворка в формат другого.
3. **Quantization** — независимый компонент, в котором определяется интерфейс для квантования весов моделей с помощью встроенных в различные фреймворки инструментов.
4. **ConfigMaker** — графическое приложение для автоматизации процедуры формирования конфигурационных файлов для разных запускаемых компонент системы.
5. **Deployment** — компонент, обеспечивающий автоматическое развертывание тестовой инфраструктуры на вычислительных узлах средствами технологии Docker. Информация об узлах содержится в конфигурационном файле компонента.
6. **RemoteController** — компонент, выполняющий удаленный запуск экспериментов для сбора показателей производительности и качества глубоких моделей на вычислительных узлах с использованием компонент **BenchmarkApp** и **AccuracyChecker** соответственно и последующую агрегацию результатов экспериментов.
7. **BenchmarkApp** — компонент, реализующий сбор показателей производительности вывода набора моделей с использованием различных инструментов глубокого обучения. Информация о моделях и параметрах вывода описывается в конфигурационном файле.
8. **AccuracyChecker** — программная обертка над аналогичным инструментом из состава OpenVINO, который обеспечивает сбор показателей качества набора моделей. Информация о моделях содержится в конфигурационном файле.
9. **ResultConverters** — вспомогательный компонент, содержащий скрипты для конвертации результатов производительности и качества, которые агрегируются компонентом **RemoteController**, в форматы html и xlxs.
10. **Inference** — компонент, содержащий реализацию вывода моделей глубокого обучения средствами различных фреймворков. Присутствует поддержка программных интерфейсов для языков C++ и Python.
11. **Tests** — компонент, обеспечивающий автоматическую проверку корректности разработанных реализаций вывода с помощью различных фреймворков. На текущий момент проверяется возможность запуска и вывода показателей производительности.
12. **DemoAPP** — консольное приложение, демонстрирующее работу системы DLI в правильной последовательности. Использует явно **BenchmarkApp**, **AccuracyChecker** и неявно оставшиеся компоненты системы.

На схеме (рис. 1) полужирными рамками выделены части компонент, которые разработаны или модифицированы в рамках выполнения настоящего исследования: конвертеры моделей для TVM и ExecuTorch (компонент **Model Converters**), реализация вывода средствами TVM (Python API) и ExecuTorch (C++ и Python APIs) (компонент **Inference**). Отметим, что также добавлены обертки для квантования весов глубоких моделей (компонент **Quantization**). Эксперименты показывают, что в настоящее время разные версии некоторых фреймворков работают нестабильно с квантованными нейросетями на устройствах RISC-V, поэтому вопросы производительности таких моделей далее не затрагиваются.

6. Вычислительные эксперименты

6.1. Показатели качества

Для оценки качества классификации используется показатель точности top-k (top-k accuracy). Пусть N — количество допустимых категорий изображений, тогда выход модели — вектор достоверностей $y^j = (y_1^j, y_2^j, ..., y_N^j)$ для каждого изображения I^j в выборке,

где $j = \overline{1,S}$, S — общее количество изображений, y_i^j — достоверность того, что изображение I^j принадлежит классу i. Если среди k наибольших достоверностей $y_{i_1^j}^j, y_{i_2^j}^j, ..., y_{i_k^j}^j$, присутствует достоверность, соответствующая искомому классу, то изображение считается проклассифицированным корректно. Тогда *точность top-k* определяется как отношение числа правильно проклассифицированных изображений к общему их количеству:

$$topk = \frac{\sum_{j=1}^{N} I(l_j \in \{i_1^j, i_2^j, ..., i_k^j\})}{S}, \tag{1}$$

где l_j — номер искомого класса, $I(l_j \in \{i_1^j, i_2^j, ..., i_k^j\})$ — индикаторная функция, которая принимает значение 1, если $l_j \in \{i_1^j, i_2^j, ..., i_k^j\}$, и 0, в противном случае. Далее в экспериментах рассматриваются точности top-1 и top-5.

6.2. Показатели производительности

Эксперимент предполагает, что набор обрабатываемых данных разбивается на *пачки (batch)* равного размера. Решение задачи классификации для пачки данных — это прямой проход по обученной нейронной сети и вычисление финального вектора достоверностей. Количество пачек, на которых запускается прямой проход, определяет число таких проходов — *итераций*. Итерации выполняются последовательно, следующая итерация запускается после завершения предыдущей. Для каждого прямого прохода измеряется продолжительность его выполнения t_i, $i = \overline{1,L}$, где L — количество итераций. Для оценки производительности вывода используется показатель *числа кадров, обрабатываемых в секунду (Frames per Second, FPS)*, который вычисляется как отношение размера входного набора данных (общего числа изображений) S к суммарному времени выполнения всех итераций:

$$FPS = \frac{S}{\sum_{i=1}^{L} t_i}. \tag{2}$$

Отметим, что время вывода с использованием разных фреймворков для одной и той же модели может отличаться в зависимости от размера входной пачки данных. Чтобы прогнозировать время завершения экспериментов, выполняются пробные запуски и для каждого размера пачки фиксируется разное количество итераций (табл. 1). Выбор максимального размера пачки обусловлен размерами оперативной памяти устройства.

Таблица 1. Выполняемое количество итераций для каждого размера входной пачки данных

Размер пачки	1	2	4	8	16	32	64	128
Количество итераций	100	100	85	70	55	40	25	20

6.3. Наборы данных

Для проведения экспериментов используется подмножество изображений валидационной выборки набора данных ImageNet [25]. Поскольку доступные на сегодняшний день образцы процессоров архитектуры RISC-V пока существенно отстают от высокопроизводительных устройств, то вывод глубоких моделей работает относительно медленно. В связи с этим для проверки точности нейронных сетей используются первые 1 000 изображений выборки. В случае корректности работы модели такой подход позволяет получить значение точности классификации, близкое к опубликованному авторами обученной сети. Для анали-

за производительности вывода выбраны 32 произвольных изображения из той же выборки. Следует отметить, что для определения показателей производительности также можно использовать синтетические входные данные, поскольку количество операций, выполняемых на прямом проходе, не зависит от контекста изображений.

6.4. Тестовые модели

Анализ производительности вывода выполняется для широко известных классификационных моделей — ResNet-50 [3] и MobileNetV2 [4]. Эти модели являются сверточными нейронными сетями с остаточными связями. Принципиальное отличие MobileNetV2 от ResNet-50 состоит в использовании сверток, отделимых по глубине. Классическая свертка предполагает проход трехмерным ядром слева направо и сверху вниз по входной трехмерной матрице (тензору) и вычисление суммы произведений соответствующих компонент. Свертка, отделимая по глубине, предусматривает последовательное применение точечной и пространственной свертки. *Точечная свертка* включает вычисление набора одномерных сверток (скалярных произведений) вдоль размерности, соответствующей каналам входной трехмерной матрицы. *Пространственная свертка* предполагает применение двумерных ядер к каждому каналу входного тензора.

В процессе анализа используются обученные модели ResNet-50 и MobileNetV2 из репозитория torchvision [26]. Модели предварительно загружаются, сериализуются и сохраняются с помощью встроенной функции ‘torch.jit.save’ библиотеки PyTorch. Затем они конвертируются и/или компилируются в форматы TensorFlow Lite, Apache TVM и ExecuTorch (рис. 2).

1. *Конвертация в формат TensorFlow Lite.* Выполняется с помощью фреймворка TensorFlow Backend for ONNX [27] в формат ONNX, далее в формат TensorFlow Lite.
2. *Конвертация в формат TVM.* Обеспечивается посредством вызова встроенной функции ‘torch.onnx.export’, которая преобразует модель в формат ONNX, после чего полученное представление передается на вход конвертеру моделей в формат TVM, реализованному в рамках системы DLI.
3. *Конвертация в формат ExecuTorch.* Реализуется в два этапа. Вначале выполняется оптимизация PyTorch-модели с использованием заданного бэкенда с помощью функции ‘to_edge_transform_and_lower’ библиотеки ExecuTorch. Далее оптимизированная модель сериализуется в формат ExecuTorch. В работе в качестве бэкенда применяется библиотека XNNPACK [28], поскольку она оптимизирована для устройств RISC-V.

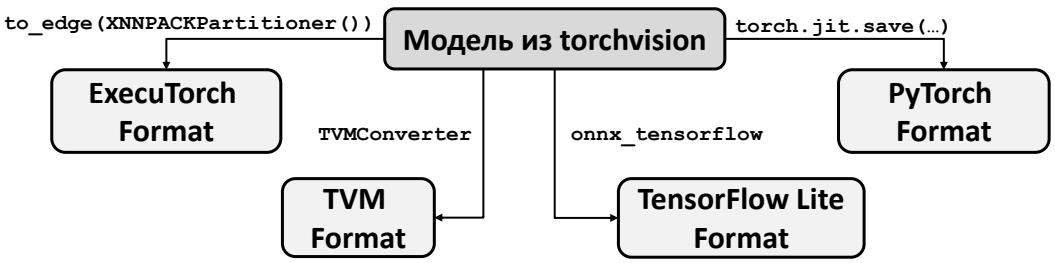

Рис. 2. Последовательность конвертации тестовых моделей в форматы целевых фреймворков

6.5. Тестовая инфраструктура

Ниже (табл. 2) приведены параметры тестовой инфраструктуры, использованной для проведения экспериментов. Фреймворки глубокого обучения собираются из исходных кодов с помощью компиляторов gcc и g++ версии 13.2.0-23ubuntu4bb3.

Таблица 2. Тестовая инфраструктура

CPU, RAM	Spacemit (R) X60, 1.6GHz, 8 ядер, 8 потоков, 16GB
Операционная система	Bianbu 2.1
Фреймворки	PyTorch v2.6.0 (OpenBLAS 0.3.26+ds-1) TensorFlow Lite v2.14.0 (XNNPACK бэкенд) Apache TVM v0.14.0 ExecuTorch v0.5.0 (XNNPACK бэкенд)

6.6. Параметры экспериментов

В табл. 3 приведены перебираемые параметры экспериментов. Вывод запускается с использованием программных интерфейсов для языков C++ и Python. При подборе оптимальных параметров для каждого размера входной пачки данных обеспечивается поиск оптимального числа потоков, которое устанавливается для обеспечения параллелизма. Отметим, что в Python API для TVM и ExecuTorch используется значение количества потоков по умолчанию, равное числу физических ядер. Также TVM позволяет установить уровень оптимизации модели opt_level в процессе ее предварительной компиляции для вывода.

Таблица 3. Параметры запуска вывода с использованием разных фреймворков

Фреймворк	**Программный интерфейс**	**Параметры**		
		Размер пачки	**Количество потоков**	**Внутренние параметры**
PyTorch	C++	+	+	—
	Python	+	+	—
TensorFlow Lite	C++	+	+	—
	Python	+	+	—
Apache TVM	Python	+	по умолчанию	уровень оптимизации opt_level
ExecuTorch	C++	+	+	—
	Python	+	по умолчанию	—

6.7. Результаты экспериментов

Качество классификации изображений. Качество решения поставленной задачи с использованием доступных фреймворков верифицируется средствами компонента **Accuracy Checker** системы DLI, который является оберткой над соответствующим инструментом в составе OpenVINO [29]. Возможности указанного инструмента расширены авторами статьи, поскольку исходная версия не поддерживает вывод глубоких моделей с помощью TVM и ExecuTorch. Реализация выложена в открытый доступ [30]. Результирующие показатели качества представлены ниже (рис. 3). Из гистограмм можно сделать вывод, что для обеих моделей полученные значения близки к опубликованным. Отличие для модели ResNet-50 по метрике top-1 составляет 0.758, по top-5 — 1.166; для модели MobileNetV2 по top-1 — 0.154, по top-5 — 0.778. Отклонения от заявленных значений точности могут

варьироваться при выборе разных подмножеств валидационной выборки ImageNet. Если распределение данных в процентном соотношении для подмножества совпадет с распределением всей валидационной выборки, то значения метрики top-1 должны быть практически равными. Также отличие может быть обусловлено тем, что некоторые фреймворки в процессе конвертации модели в собственное внутреннее представление изменяют порядок выполнения операций, в связи с чем на данных, где модель дает близкие выходные значения достоверностей, может изменяться порядок классов при вычислении top-1 или top-5.

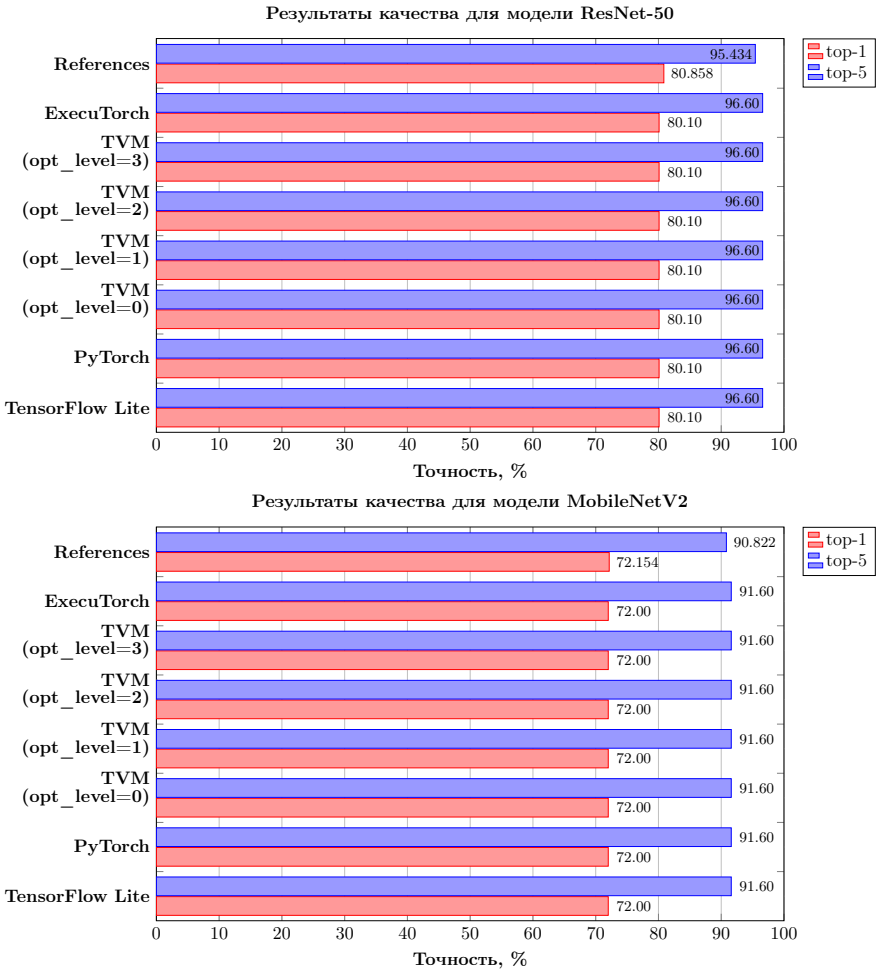

Рис. 3. Качество классификации на множестве из первых 1 000 изображений валидационной выборки ImageNet

Подбор оптимальных параметров. На данном этапе для каждого допустимого размера входной пачки данных необходимо подобрать оптимальные параметры запуска. Фреймворк *PyTorch* в программных интерфейсах C++ и Python позволяет перебирать количество потоков. Из полученных результатов (рис. 4) можно видеть, что для обеих тестовых моделей поведение показателя производительности нестабильно. Для модели ResNet-50 лучшие значения показателя FPS достигаются на размере пачки, равном 1, как для программного интерфейса языка C++, так и для Python. Отличие во втором знаке после запятой ($\sim$0.044). Такой сценарий получения входных данных характерен для многих приложений, обеспечивающих обработку видео с низкой частотой, либо не требующих запуска вывода глубокой модели на каждом кадре видеопотока. Для MobileNetV2 ситуация кардинально противоположная. В целом с увеличением размера пачки повышается про-

изводительность, при этом максимальная — наблюдается на наибольшем размере входной пачки изображений. Согласно открытым источникам (официальный форум [31]) PyTorch собирается для запуска на устройствах RISC-V, но фреймворк не оптимизирован под данную архитектуру. К сожалению, в доступных процессорах архитектуры RISC-V не хватает привычных для x86-64 датчиков производительности, и средства профилировки обладают скромными возможностями. Поэтому сложно указать точные причины подобных артефактов. Следует отметить, что анализ результатов производительности вывода на устройствах x86-64 для тех же тестовых моделей показывает хорошую масштабируемость.

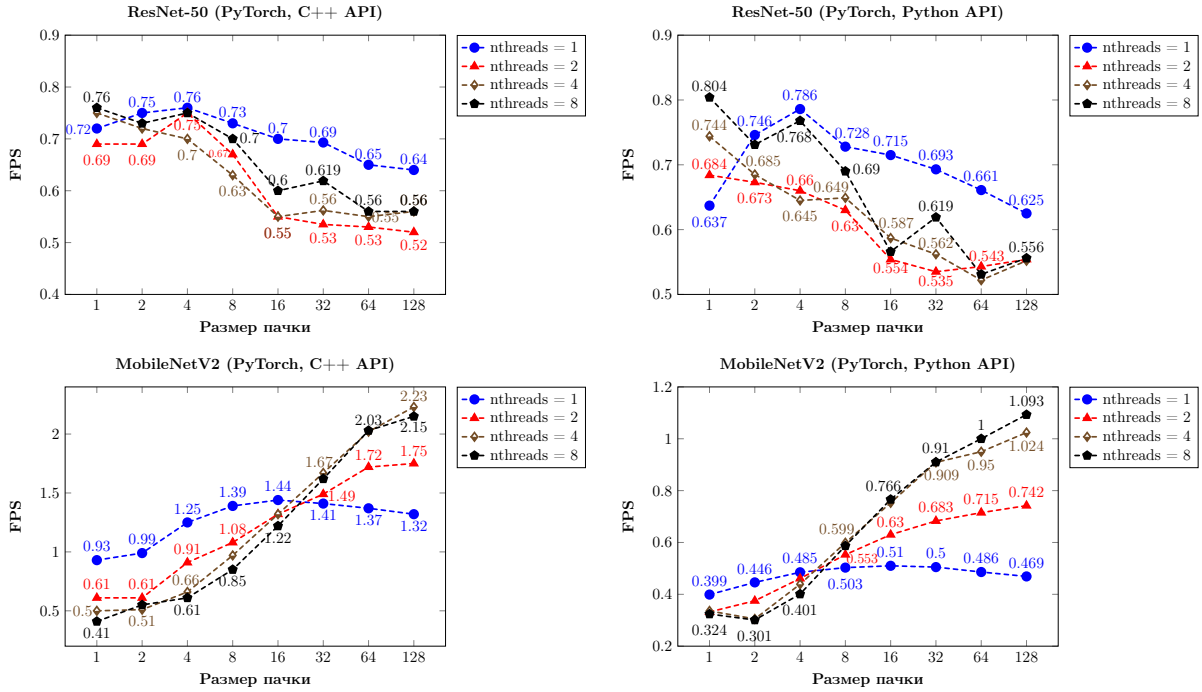

Рис. 4. Зависимости числа кадров, обрабатываемых за секунду, от размера входной пачки данных при разном количестве потоков для фреймворка PyTorch. Каждый график соответствует определенной тестовой модели, запущенной с использованием C++ или Python API

TensorFlow Lite по аналогии с PyTorch позволяет перебирать количество потоков для параллельного исполнения вывода. Результаты экспериментов (рис. 5) показывают хорошую масштабируемость с ростом числа потоков для модели ResNet-50. При этом увеличение размера пачки данных не приводит к падению производительности. Поэтому выбор оптимального размера пачки в основном зависит от скорости получения входных данных в приложении.

Для MobileNetV2 ситуация несколько хуже, в особенности при использовании максимального числа потоков на малых размерах пачки данных (менее четырех изображений). На пачках, размер которых превышает 4, проблемы с масштабируемостью постепенно уходят, но начиная с 32 изображений происходит спад. При этом на 16 изображениях увеличение числа потоков вдвое приводит к ускорению, близкому к 1.9. Предположительно такой результат обусловлен особенностями параллельной реализации пространственных сверток, которые составляют основное преобразование в сети MobileNetV2 и требуют нерегулярного обхода тензоров. Таким образом, при запуске указанной модели имеет смысл подавать на

вход пачку данных размера 16 и устанавливать число потоков, равное 8. Если же приложение не позволяет сформировать пачку, размер которой превышает 2 изображения, то при параллельном исполнении следует задействовать только 4 потока.

Отдельно следует отметить, что Python API в целом демонстрирует скорость обработки изображений близкую или немного ниже по сравнению с C++ API для обеих тестовых моделей вследствие наличия накладных расходов на вызов функций библиотеки C++.

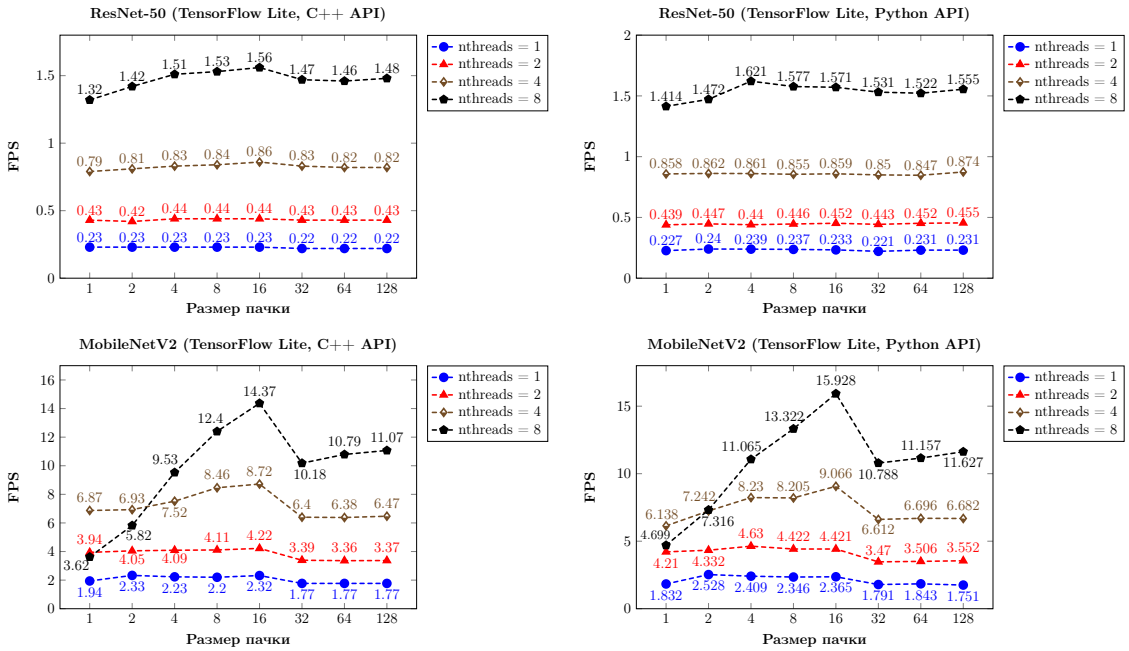

Рис. 5. Зависимости числа кадров, обрабатываемых за секунду, от размера входной пачки данных при разном количестве потоков для фреймворка TensorFlow Lite. Каждый график соответствует определенной тестовой модели, запущенной с использованием C++ или Python API

TVM позволяет установить уровень оптимизации модели. Ниже приведены (рис. 6) графики зависимости показателя числа обрабатываемых за секунду кадров от размера входной пачки данных. Теоретически чем выше уровень оптимизации (opt_level), тем быстрее должен работать вывод. На практике это утверждение не всегда справедливо. Для модели ResNet-50, например, на пачках 2, 4 и 8 показатели производительности отличаются незначительно (второй знак после запятой), а для MobileNetV2 на всех размерах пачки, за исключением 1 и 128, при opt_level=3 наблюдаются лучшие значения FPS. Отсутствие эффекта от оптимизации модели ResNet-50 объясняется тем, что основная операция в сети — это классическая свертка, которая изначально хорошо оптимизирована во фреймворке. Небольшое улучшение производительности вывода MobileNetV2 при изменении уровня оптимизации, вероятнее всего, связано с применением более эффективных стратегий обхода входных тензоров при реализации сверток, отделимых по глубине. При этом следует отметить «провал» производительности на пачке в 32 изображения и opt_level=1, возникший предположительно из-за простоев при работе с памятью. Таким образом, для первой тестовой модели при каждом размере пачки нельзя однозначно рекомендовать оптимальное значение параметра opt_level, в то время как для второй — на больших размерах пачки максимальный уровень оптимизации гарантирует лучшую производительность вывода.

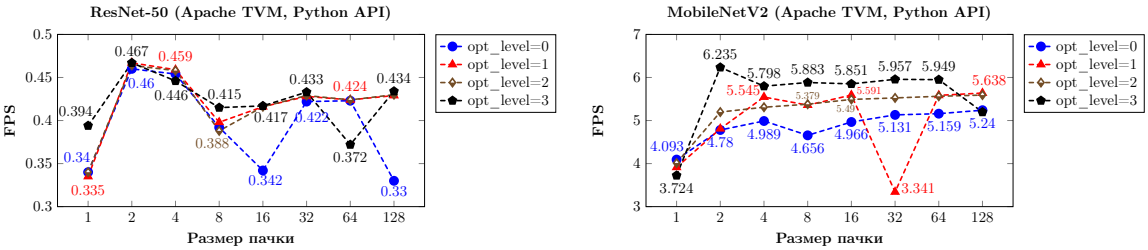

Рис. 6. Зависимости числа кадров, обрабатываемых за секунду, от размера входной пачки данных при разном уровне оптимизации модели (opt_level) для фреймворка Apache TVM

ExecuTorch позволяет перебирать количество потоков для параллельного запуска вывода только в C++ API, в Python API данный параметр устанавливается в значение по умолчанию, равное количеству физических ядер. Ниже приведены результаты определения оптимального числа потоков для C++ API (рис. 7) и сравнение лучших результатов C++ API с Python API для каждого допустимого размера пачки данных (рис. 8).

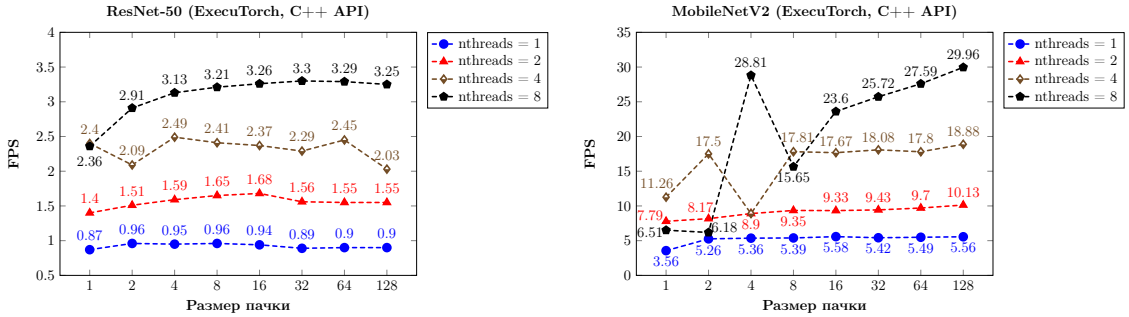

Рис. 7. Зависимости числа кадров, обрабатываемых за секунду, от размера входной пачки данных для фреймворка ExecuTorch (C++ API)

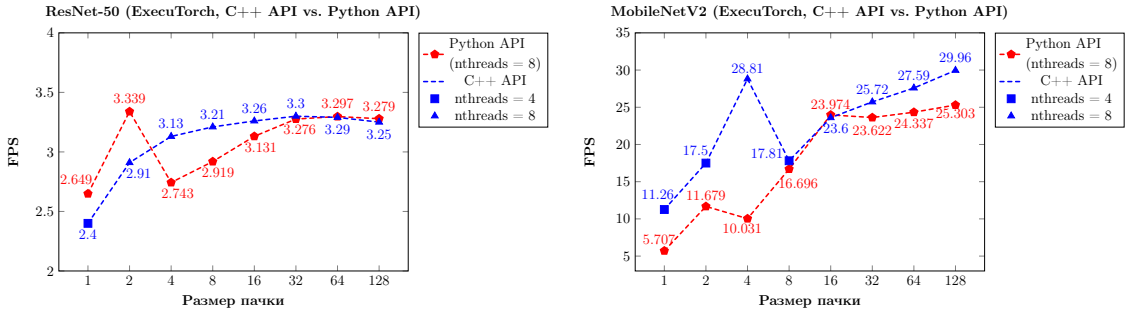

Рис. 8. Сравнение результатов производительности вывода для Python и C++ APIs фреймворка ExecuTorch

Для модели ResNet-50 фреймворк демонстрирует рост производительности при увеличении числа потоков независимо от размера входной пачки данных (рис. 7, слева). Так для пачки в 32 изображения, на которой достигается максимальная производительность, с увеличением числа потоков вдвое ускорение в среднем составляет ∼1.55 раза. При этом для каждого фиксированного числа потоков с увеличением размера пачки данных FPS либо не изменяется, либо растет (во втором знаке после запятой). Для сети MobileNetV2

при запуске в 2 потока также наблюдается увеличение FPS в среднем ∼1.77 раза, а на 4 и 8 потоках (рис. 7, справа) — нестабильное поведение показателя производительности. При запуске в 4 потока очевиден «провал» на пачке в 4 изображения, а при запуске в 8 потоков на размере пачки, равном 8, возникает пик в 28.81 fps, близкий к максимальному значению производительности (29.96 fps на пачке 128). Как отмечалось ранее, в настоящее время на RISC-V отсутствуют технические возможности, чтобы достоверно объяснить причины недостаточного ускорения от параллелизма и других артефактов. Предположительно это связано с тем, что плата Banana Pi BPI-F3 оснащена не самой быстрой подсистемой памяти, также в процессоре отсутствует внеочередное исполнение инструкций, в результате чего меньше возможностей для уменьшения простоев из-за работы с памятью. Пик на 8 потоках вероятнее всего достигается вследствие удачного расположения данных в памяти. Подробнее подобные вопросы обсуждаются в [32].

Если сравнивать лучшие результаты производительности, полученные при использовании C++ API, с показателями Python API (рис. 8), то можно сделать следующие выводы. Для модели ResNet-50 показатели в целом отличаются незначительно, Python API на большинстве размеров входных пачек уступает из-за наличия накладных расходов на вызов функций библиотеки C++. Аналогичное утверждение справедливо и для модели MobileNetV2, но эта разница выглядит более существенной для размеров пачки 1, 2 и 4. В связи с этим в приложениях предпочтительно использовать для вывода C++ API. Обе тестовые модели имеет смысл запускать на максимально возможном размере пачки данных 128 в 8 потоков.

Сравнительный анализ. Сравнение показателей производительности, полученных при оптимальных параметрах запуска вывода (рис. 9), показывает, что ExecuTorch обеспечивает лучшую скорость обработки изображений независимо от размера входной пачки данных. Для модели ResNet-50 скорость обработки меняется от 2.649 до 3.339 fps при оптимальных параметрах запуска в зависимости от размера входной пачки данных, для MobileNetV2 — от 11.26 до 29.96 fps. Вывод средствами TensorFlow Lite уступает ExecuTorch в среднем в ∼2.1 раза. Допускаем, что использование более новой версии библиотеки TensorFlow Lite позволит улучшить показатели производительности. Остальные фреймворки показывают существенно худшую производительность. Предположительно данный факт связан с тем, что хотя для PyTorch и Apache TVM официально заявлена поддержка RISC-V, но они не в полной мере оптимизированы под указанную архитектуру.

Заключение

Активное применение глубокого обучения для решения прикладных задач неизбежно поднимает вопросы, связанные с анализом производительности вывода моделей на конечных устройствах, которые используются для многократного прямого прохода. Темпы развития архитектуры RISC-V и обзор различных источников свидетельствуют об интересе сообщества к подобным вопросам, что говорит об актуальности результатов исследования.

В работе для классификационных моделей ResNet-50 и MobileNetV2 демонстрируется, что среди известных фреймворков глубокого обучения, обеспечивающих запуск вывода на устройствах RISC-V, на текущий момент на плате Banana Pi BPI-F3 лучшую производительность показывает фреймворк ExecuTorch (с XNNPACK-бэкендом). Для сети ResNet-50, где преобладают классические свертки с трехмерным ядром и одномерные свертки вдоль размерности, соответствущей каналам, наблюдается хорошая масштабируемость независи-

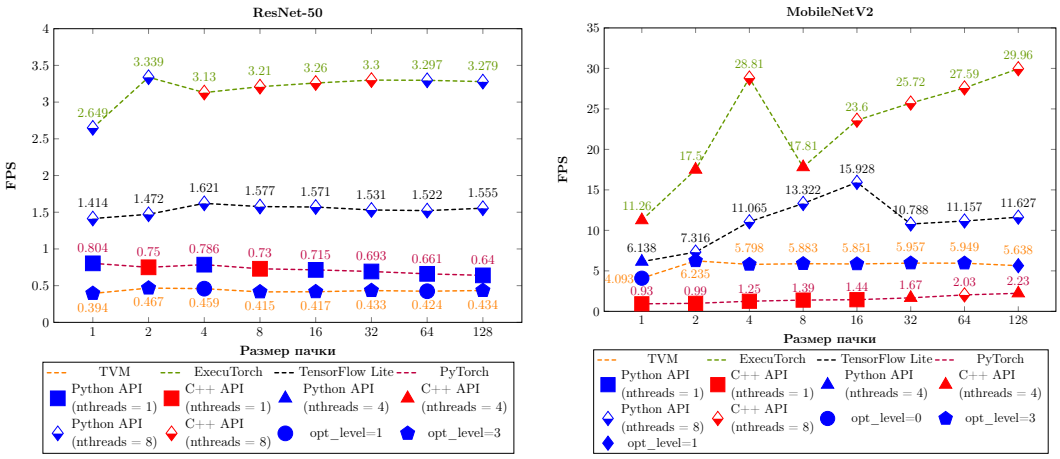

Рис. 9. Лучшие результаты производительности, полученные при использовании оптимальных параметров запуска вывода. Цвет и тип линий соответствует фреймворку, маркеры — программному интерфейсу и оптимальным параметрам запуска

мо от размера входной пачки данных. Лучшие показатели производительности для обеих тестовых моделей достигаются при запуске в 8 потоков. При этом MobileNetV2 на пачках 4, 32, 64 и 128 изображений работает в режиме реального времени (более 25 fps). На практике выбор оптимального размера входной пачки данных зависит не только от результатов анализа производительности, но от скорости поступления входных данных в конкретном приложении (изображений в системах видеоанализа).

Несмотря на то, что Banana Pi BPI-F3 относится к числу маломощных, вывод нейронных сетей на таких устройствах имеет смысл. Эксперименты показывают, что даже на классическом примере известной «легковесной» сети MobileNetV2 удается обеспечить обработку данных в режиме реального времени. Данный факт говорит в поддержку тезиса о перспективности внедрения процессоров RISC-V в мобильные устройства. Полученные результаты исследования позволяют сделать вывод, что при развитии архитектуры RISC-V полезно обратить внимание на возможность аппаратного ускорения операции свертки, поскольку она является наиболее вычислительно-трудоемкой для большого класса сетей.

Разработанная инфраструктура в рамках программной системы Deep Learning Inference Benchmark позволяет автоматизировать запуск вывода и агрегацию результатов бенчмаркинга вывода глубоких моделей. Система является открытой и расширяемой, поэтому может быть использована для анализа производительности вывода в аналогичных исследованиях на новых образцах процессоров архитектуры RISC-V.

Литература

1. Noor M.H.M., Ige A.O. A survey on state-of-the-art deep learning applications and challenges // Engineering Applications of Artificial Intelligence. 2025. Vol. 159, Part B. P. 111225. DOI: 10.1016/j.engappai.2025.111225.

2. Mezger B.W., *et al.* A Survey of the RISC-V Architecture Software Support // IEEE Access. 2022. Vol. 10. P. 51394–51411. DOI: 10.1109/ACCESS.2022.3174125.

3. He K., Zhang X., Ren S., Sun J. Deep Residual Learning for Image Recognition // 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 2016. P. 770–778. DOI: 10.1109/CVPR.2016.90.

4. Sandler M., *et al.* MobileNetV2: Inverted Residuals and Linear Bottlenecks // 2018 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Salt Lake City, UT, USA, June 18–22, 2018. P. 4510–4520. DOI: 10.1109/CVPR.2018.00474.

5. Алибеков М.Р. и др. Методика анализа производительности вывода глубоких нейронных сетей на примере задачи классификации изображений // Вычислительные методы и программирование. 2024. Т. 25, № 2. С. 127–141. DOI: 10.26089/NumMet.v25r211.

6. Demidovskij A., *et al.* OpenVINO Deep Learning Workbench: Comprehensive Analysis and Tun-ing of Neural Networks Inference // ICCV Workshop, 2019. URL: `https://openaccess.thecvf.com/content_ICCVW_2019/papers/SDL-CV/Gorbachev_OpenVINO_Deep_Learning_Workbench_Comprehensive_Analysis_and_Tuning_of_Neural_ICCVW_2019_paper.pdf` (дата обращения: 25.07.2025).

7. Arya M., Simmhan Y. A Preliminary Performance Analysis of LLM Inference on Edge Accelerators // 2024 IEEE 31st International Conference on High Performance Computing, Data and Analytics Workshop (HiPCW), Bangalore, India, 2024. P. 183–184. DOI: 10.1109/HiPCW63042.2024.00069.

8. Verma G., *et al.* Performance Evaluation of Deep Learning Compilers for Edge Inference // 2021 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW), Portland, OR, USA, 2021. P. 858–865. DOI: 10.1109/IPDPSW52791.2021.00128.

9. Chen Y.-R., *et al.* Experiments and optimizations for TVM on RISC-V Architectures with P Extension // 2020 International Symposium on VLSI Design, Automation and Test (VLSI-DAT), Hsinchu, Taiwan, 2020. P. 1–4. DOI: 10.1109/VLSI-DAT49148.2020.9196477.

10. Christofas V., *et al.* Comparative Evaluation between Accelerated RISC-V and ARM AI Inference Machines // 2023 6th World Symposium on Communication Engineering (WSCE), Thessaloniki, Greece, 2023. P. 108–113. DOI: 10.1109/WSCE59557.2023.10365853.

11. Bhattacharjee D., *et al.* Full-Stack Evaluation of Machine Learning Inference Workloads for RISC-V Systems // RISC-V Summit 2024. URL: `https://riscv-europe.org/summit/2024/media/proceedings/posters/58_poster.pdf` (дата обращения: 12.10.2025).

12. Garcia A.M., *et al.* Assessing Large Language Models Inference Performance on a 64-core RISC-V CPU with Silicon-Enabled Vectors // BigHPC2024: Special Track on Big Data and High-Performance Computing, co-located with the 3rd Italian Conference on Big Data and Data Science, ITADATA2024, Pisa, Italy, September 17–19, 2024. URL: `https://ceur-ws.org/Vol-3785/paper110.pdf` (дата обращения: 19.07.2025).

13. Martinez H., *et al.* Performance Analysis of BERT on RISC-V Processors with SIMD Units // High Performance Computing. ISC High Performance 2024 International Workshops. Vol. 15058 / eds. by M. Weiland, S. Neuwirth, C. Kruse, T. Weinzierl. Springer, 2024. P. 325–338. Lecture Notes in Computer Science. DOI: 10.1007/978-3-031-73716-9_23.

14. Suárez D., *et al.* Energy-Efficient Inference on RNN and LLM networks: A Quantized Evaluation on RISC-V, ARM, and x86 Devices // 16th ACM International Conference on Future and Sustainable Energy Systems (E-Energy '25). Association for Computing Machinery, New York, NY, USA, 2025. P. 882–889. DOI: 10.1145/3679240.3735101.

15. Mukhin I., *et al.* Benchmarking Deep Learning Inference on RISC-V CPUs // Supercomputing. RuSCDays 2024. Vol. 15406 / eds. by V. Voevodin, A. Antonov, D. Nikitenko. Springer, 2025. P. 331–346. Lecture Notes in Computer Science. DOI: 10.1007/978-3-031-78459-0_24.

16. Официальная страница фреймворка PyTorch. URL: `https://pytorch.org` (дата обращения: 19.07.2025).

17. Официальная страница фреймворка TensorFlow Lite. URL: `https://www.tensorflow.org/lite` (дата обращения: 19.07.2025).

18. Chen T., *et al.* TVM: An Automated End-to-End Optimizing Compiler for Deep Learning // 13th USENIX Conference on Operating Systems Design and Implementation, Carlsbad, CA, USA, 2018. P. 579–594. DOI: 10.5555/3291168.3291211.

19. Официальная страница фреймворка ExecuTorch. URL: `https://pytorch.org/executorch-overview` (дата обращения: 19.07.2025).

20. DLI: Deep Learning Inference Benchmark. Репозиторий проекта. URL: `https://github.com/itlab-vision/dl-benchmark` (дата обращения: 19.07.2025).

21. Kustikova V., *et al.* DLI: Deep Learning Inference Benchmark // Supercomputing. RuSCDays 2019. Vol. 1129 / eds. by V. Voevodin, S. Sobolev. Springer, 2019. P. 542–553. Communications in Computer and Information Science. DOI: 10.1007/978-3-030-36592-9_44.

22. Lin D., *et al.* Fixed Point Quantization of Deep Convolutional Networks // 33rd International Conference on Machine Learning, New York, USA, 2016. P. 2849–2858. URL: `https://proceedings.mlr.press/v48/linb16.pdf` (дата обращения: 19.07.2025).

23. Kozlov A., *et al.* Neural network compression framework for fast model inference // Intelligent Computing 2021. Vol. 285 / ed. by K. Arai. Springer, 2021. P. 213–232. Lecture Notes in Networks and Systems. DOI: 10.1007/978-3-030-80129-8_17.

24. Официальная страница фреймворка ncnn. URL: `https://github.com/Tencent/ncnn` (дата обращения: 19.07.2025).

25. Deng J., *et al.* ImageNet: A Large-Scale Hierarchical Image Database // IEEE Computer Vision and Pattern Recognition (CVPR), 2009. URL: `https://www.image-net.org/static_files/papers/imagenet_cvpr09.pdf` (дата обращения: 19.07.2025).

26. TorchVision: PyTorch's Computer Vision library. 2016. URL: `https://github.com/pytorch/vision` (дата обращения: 19.07.2025).

27. TensorFlow Backend for ONNX. URL: `https://github.com/onnx/onnx-tensorflow` (дата обращения: 19.07.2025).

28. XNNPACK. High-efficiency floating-point neural network inference operators for mobile, server, and Web. URL: `http://github.com/google/XNNPACK` (дата обращения: 19.07.2025).

29. OpenVINO. OpenVINO is an open source toolkit for optimizing and deploying AI inference. URL: `https://github.com/openvinotoolkit/openvino` (дата обращения: 19.07.2025).

30. OpenVINO Toolkit — Open Model Zoo repository. Fork (branch '24.3.0/tvm' for Apache TVM, branch 'omz_executorch' for ExecuTorch). URL: `https://github.com/itlab-vision/open_model_zoo_tvm` (дата обращения: 19.07.2025).

31. PyTorch RISC-V support. URL: `https://discuss.pytorch.org/t/pytorch-risc-v-support/212065` (дата обращения: 19.07.2025).

32. Pirova A., *et al.* Performance optimization of BLAS algorithms with band matrices for RISC-V processors // Future Generation Computer Systems. 2026. Vol. 174. P. 107936. DOI: 10.1016/j.future.2025.107936.

Мухин Иван Сергеевич, студент, кафедра высокопроизводительных вычислений и системного программирования, Институт информационных технологий, математики и механики, Нижегородский государственный университет им. Н.И. Лобачевского (Нижний Новгород, Российская Федерация)

Кустикова Валентина Дмитриевна, к.т.н., доцент, кафедра высокопроизводительных вычислений и системного программирования, Институт информационных технологий, математики и механики, Нижегородский государственный университет им. Н.И. Лобачевского (Нижний Новгород, Российская Федерация)

DOI: 10.14529/cmse250403

PERFORMANCE ANALYSIS OF DEEP LEARNING INFERENCE ON THE BANANA PI BPI-F3 BOARD USING THE IMAGE CLASSIFICATION PROBLEM AS AN EXAMPLE

© 2025 I.S. Mukhin, V.D. Kustikova

Lobachevsky State University of Nizhny Novgorod

(pr. Gagarina 23, Nizhny Novgorod, 603022 Russia)

E-mail: ismukhin03@gmail.com, valentina.kustikova@itmm.unn.ru

Received: 18.08.2025

The paper analyzes the inference performance of the well-known neural networks ResNet-50 and MobileNetV2, which provide a solution for the problem of image classification, on the Banana Pi BPI-F3 board, which is built on the RISC-V architecture. The inference is launched by available frameworks: PyTorch, TensorFlow Lite, Apache TVM and ExecuTorch. The models are converted to the format of each target framework. The correctness of the problem solving is checked using the obtained neural networks. It is demonstrated that the accuracy indicators of image classification using these models correlate well with the published ones. Then, the optimal parameters for launching the inference for each framework and model are selected. A comparative analysis of the inference performance shows that ExecuTorch demonstrates the best results for both models. For the ResNet-50 model, the number of frames processed per second (FPS) varies from 2.649 to 3.339 fps with optimal parameters depending on the batch size of images processed in one forward pass through the network, for MobileNetV2 – from 11.26 to 29.96 fps. TensorFlow Lite is inferior to ExecuTorch by an average of ∼2.1 times. PyTorch and Apache TVM demonstrate lower performance indicators. Probably, this is due to the fact that they are not fully optimized for the RISC-V architecture.

Keywords: deep learning, image classification, inference performance, PyTorch, TensorFlow Lite, Apache TVM, ExecuTorch, Banana Pi BPI-F3, RISC-V.

FOR CITATION

Mukhin I.S., Kustikova V.D. Performance Analysis of Deep Learning Inference on the Banana Pi BPI-F3 Board Using the Image Classification Problem as an Example. Bulletin of the South Ural State University. Series: Computational Mathematics and Software Engineering. 2025. Vol. 14, no. 4. P. 40–60. (in Russian) DOI: 10.14529/cmse250403.

References

1. Noor M.H.M., Ige A.O. A survey on state-of-the-art deep learning applications and challenges. Engineering Applications of Artificial Intelligence. 2025. Vol. 159, Part B. P. 111225. DOI: 10.1016/j.engappai.2025.111225.

2. Mezger B.W., *et al.* A Survey of the RISC-V Architecture Software Support. IEEE Access. 2022. Vol. 10. P. 51394–51411. DOI: 10.1109/ACCESS.2022.3174125.

3. He K., Zhang X., Ren S., Sun J. Deep Residual Learning for Image Recognition. 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 2016. P. 770–778. DOI: 10.1109/CVPR.2016.90.

4. Sandler M., *et al.* MobileNetV2: Inverted Residuals and Linear Bottlenecks. 2018 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Salt Lake City, UT, USA, June 18–22, 2018. P. 4510–4520. DOI: 10.1109/CVPR.2018.00474.

5. Alibekov M.R., *et al.* Performance analysis methodology of deep neural networks inference on the example of an image classification problem. Numerical Methods and Programming. 2024. Vol. 25, no. 2. P. 127–141. (in Russian) DOI: 10.26089/NumMet.v25r211.

6. Demidovskij A., *et al.* OpenVINO Deep Learning Workbench: Comprehensive Analysis and Tun-ing of Neural Networks Inference. ICCV Workshop, 2019. URL: `https://openaccess.thecvf.com/content_ICCVW_2019/papers/SDL-CV/Gorbachev_OpenVINO_Deep_Learning_Workbench_Comprehensive_Analysis_and_Tuning_of_Neural_ICCVW_2019_paper.pdf` (accessed: 25.07.2025).

7. Arya M., Simmhan Y. A Preliminary Performance Analysis of LLM Inference on Edge Accelerators. 2024 IEEE 31st International Conference on High Performance Computing, Data and Analytics Workshop (HiPCW), Bangalore, India, 2024. P. 183–184. DOI: 10.1109/HiPCW63042.2024.00069.

8. Verma G., *et al.* Performance Evaluation of Deep Learning Compilers for Edge Inference. 2021 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW), Portland, OR, USA, 2021. P. 858–865. DOI: 10.1109/IPDPSW52791.2021.00128.

9. Chen Y.-R., *et al.* Experiments and optimizations for TVM on RISC-V Architectures with P Extension. 2020 International Symposium on VLSI Design, Automation and Test (VLSI-DAT), Hsinchu, Taiwan, 2020. P. 1–4. DOI: 10.1109/VLSI-DAT49148.2020.9196477.

10. Christofas V., *et al.* Comparative Evaluation between Accelerated RISC-V and ARM AI Inference Machines. 2023 6th World Symposium on Communication Engineering (WSCE), Thessaloniki, Greece, 2023. P. 108–113. DOI: 10.1109/WSCE59557.2023.10365853.

11. Bhattacharjee D., *et al.* Full-Stack Evaluation of Machine Learning Inference Workloads for RISC-V Systems. RISC-V Summit 2024. URL: `https://riscv-europe.org/summit/2024/media/proceedings/posters/58_poster.pdf` (accessed: 12.10.2025).

12. Garcia A.M., *et al.* Assessing Large Language Models Inference Performance on a 64-core RISC-V CPU with Silicon-Enabled Vectors. BigHPC2024: Special Track on Big Data and High-Performance Computing, co-located with the 3rd Italian Conference on Big Data and Data Science, ITADATA2024, Pisa, Italy, September 17–19, 2024. URL: `https://ceur-ws.org/Vol-3785/paper110.pdf` (accessed: 19.07.2025).

13. Martinez H., *et al.* Performance Analysis of BERT on RISC-V Processors with SIMD Units. High Performance Computing. ISC High Performance 2024 International Workshops. Vol. 15058 / eds. by M. Weiland, S. Neuwirth, C. Kruse, T. Weinzierl. Springer, 2024. P. 325–338. Lecture Notes in Computer Science. DOI: 10.1007/978-3-031-73716-9_23.

14. Suarez D., *et al.* Energy-Efficient Inference on RNN and LLM networks: A Quantized Evaluation on RISC-V, ARM, and x86 Devices. 16th ACM International Conference on Future and Sustainable Energy Systems (E-Energy '25). Association for Computing Machinery, New York, NY, USA, 2025. P. 882–889. DOI: 10.1145/3679240.3735101.

15. Mukhin I., *et al.* Benchmarking Deep Learning Inference on RISC-V CPUs. Supercomputing. RuSCDays 2024. Vol. 15406 / eds. by V. Voevodin, A. Antonov, D. Nikitenko. Springer, 2025. P. 331–346. Lecture Notes in Computer Science. DOI: 10.1007/978-3-031-78459-0_24.

16. The official web site of the framework PyTorch. URL: `https://pytorch.org` (accessed: 19.07.2025).

17. The official web site of the framework TensorFlow Lite. URL: `https://www.tensorflow.org/lite` (accessed: 19.07.2025).

18. Chen T., *et al.* TVM: An Automated End-to-End Optimizing Compiler for Deep Learning. 13th USENIX Conference on Operating Systems Design and Implementation, Carlsbad, CA, USA, 2018. P. 579–594. DOI: 10.5555/3291168.3291211.

19. The official web site of the framework ExecuTorch. URL: `https://pytorch.org/executorch-overview` (accessed: 19.07.2025).

20. DLI: Deep Learning Inference Benchmark. GitHub Repo. URL: `https://github.com/itlab-vision/dl-benchmark` (accessed: 19.07.2025).

21. Kustikova V., *et al.* DLI: Deep Learning Inference Benchmark. Supercomputing. RuSCDays 2019. Vol. 1129 / eds. by V. Voevodin, S. Sobolev. Springer, 2019. P. 542–553. Communications in Computer and Information Science. DOI: 10.1007/978-3-030-36592-9_44.

22. Lin D., *et al.* Fixed Point Quantization of Deep Convolutional Networks. 33rd International Conference on Machine Learning, New York, USA, 2016. P. 2849–2858. URL: `https://proceedings.mlr.press/v48/linb16.pdf` (accessed: 19.07.2025).

23. Kozlov A., *et al.* Neural network compression framework for fast model inference. Intelligent Computing 2021. Vol. 285 / ed. by K. Arai. Springer, 2021. P. 213–232. Lecture Notes in Networks and Systems. DOI: 10.1007/978-3-030-80129-8_17.

24. The official web site of the framework ncnn. URL: `https://github.com/Tencent/ncnn` (accessed: 19.07.2025).

25. Deng J., *et al.* ImageNet: A Large-Scale Hierarchical Image Database. IEEE Computer Vision and Pattern Recognition (CVPR), 2009. URL: `https://www.image-net.org/static_files/papers/imagenet_cvpr09.pdf` (accessed: 19.07.2025).

26. TorchVision: PyTorch's Computer Vision library. 2016. URL: `https://github.com/pytorch/vision` (accessed: 19.07.2025).

27. TensorFlow Backend for ONNX. URL: `https://github.com/onnx/onnx-tensorflow` (accessed: 19.07.2025).

28. XNNPACK. High-efficiency floating-point neural network inference operators for mobile, server, and Web. URL: `http://github.com/google/XNNPACK` (accessed: 19.07.2025).

29. OpenVINO. OpenVINO is an open source toolkit for optimizing and deploying AI inference. URL: `https://github.com/openvinotoolkit/openvino` (accessed: 19.07.2025).

30. OpenVINO Toolkit – Open Model Zoo repository. Fork (branch '24.3.0/tvm' for Apache TVM, branch 'omz_executorch' for ExecuTorch). URL: `https://github.com/itlab-vision/open_model_zoo_tvm` (accessed: 19.07.2025).

31. PyTorch RISC-V support. URL: `https://discuss.pytorch.org/t/pytorch-risc-v-support/212065` (accessed: 19.07.2025).

32. Pirova A., *et al.* Performance optimization of BLAS algorithms with band matrices for RISC-V processors. Future Generation Computer Systems. 2026. Vol. 174. P. 107936. DOI: 10.1016/j.future.2025.107936.

СВЕДЕНИЯ ОБ ИЗДАНИИ

Научный журнал «Вестник ЮУрГУ. Серия «Вычислительная математика и информатика» основан в 2012 году.

Учредитель — Федеральное государственное автономное образовательное учреждение высшего образования «Южно-Уральский государственный университет» (национальный исследовательский университет).

Главный редактор — Л.Б. Соколинский.

Свидетельство о регистрации ПИ ФС77-57377 выдано 24 марта 2014 г. Федеральной службой по надзору в сфере связи, информационных технологий и массовых коммуникаций.

Журнал включен в Реферативный журнал и Базы данных ВИНИТИ; индексируется в библиографической базе данных РИНЦ. Журнал размещен в открытом доступе на Всероссийском математическом портале MathNet. Сведения о журнале ежегодно публикуются в международной справочной системе по периодическим и продолжающимся изданиям «Ulrich's Periodicals Directory».

Решением Президиума Высшей аттестационной комиссии Министерства образования и науки Российской Федерации журнал включен в «Перечень рецензируемых научных изданий, в которых должны быть опубликованы основные научные результаты на соискание ученой степени кандидата наук, на соискание ученой степени доктора наук» по научным специальностям и соответствующим им отраслям науки: 1.2.3 – Теоретическая информатика, кибернетика (физико-математические науки), 2.3.5 – Математическое и программное обеспечение вычислительных машин, комплексов и компьютерных сетей (физико-математические науки).

Подписной индекс научного журнала «Вестник ЮУрГУ», серия «Вычислительная математика и информатика»: 10244, каталог «Пресса России». Периодичность выхода — 4 выпуска в год.

Адрес редакции: 454080, г. Челябинск, ул. С. Кривой, 79, Издательский центр ЮУрГУ, каб. 2.

Адрес издателя: 454080, г. Челябинск, проспект Ленина, 76.

ПРАВИЛА ДЛЯ АВТОРОВ

1. Правила подготовки рукописей и пример оформления статей можно загрузить с сайта серии `https://vestnikvmi.susu.ru`. **Статьи, оформленные без соблюдения правил, к рассмотрению не принимаются.**
2. Адрес редакционной коллегии научного журнала «Вестник ЮУрГУ», серия «Вычислительная математика и информатика»:

   Россия 454080, г. Челябинск, пр. им. В.И. Ленина, 76, ЮУрГУ, НОЦ ИИКТ,
   зам. главного редактора Цымблеру М.Л.
3. Адрес электронной почты редакции: `vestnikvmi@susu.ru`.
4. **Плата с авторов за публикацию рукописей не взимается, гонорары авторам не выплачиваются.**

ВЕСТНИК
ЮЖНО-УРАЛЬСКОГО
ГОСУДАРСТВЕННОГО УНИВЕРСИТЕТА
Серия
«ВЫЧИСЛИТЕЛЬНАЯ МАТЕМАТИКА И ИНФОРМАТИКА»
2025 Том 14, № 4

16+

Техн. редактор *А.В. Миних*

Издательский центр Южно-Уральского государственного университета

Подписано в печать 08.12.2025. Дата выхода в свет 29.12.2025. Формат 60×84 1/8. Печать цифровая.
Усл. печ. л. 7,44. Тираж 500 экз. Заказ 305/331. Цена свободная.

Отпечатано в типографии Издательского центра ЮУрГУ.
454080, г. Челябинск, проспект Ленина, 76.