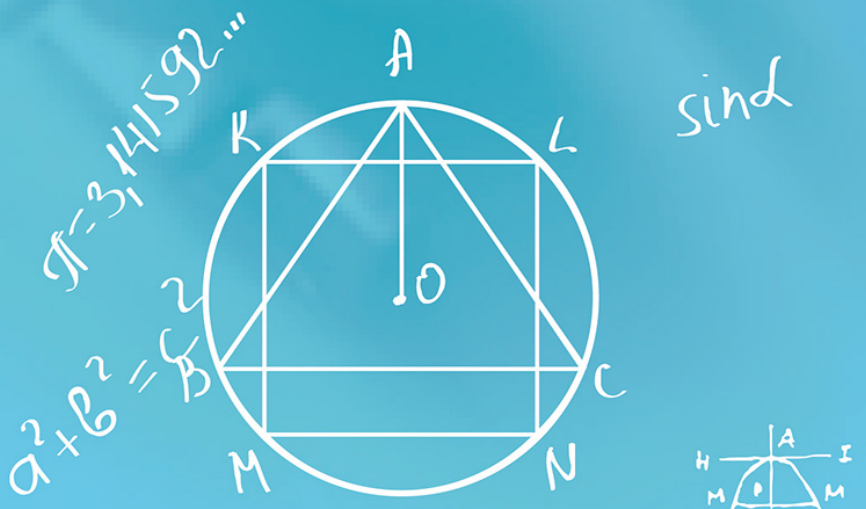
www.aurora-group.eu
www.nbpublish.com

ПРОГРАММНЫЕ СИСТЕМЫ

и вычислительные методы

научный журнал



AURORA Group s.r.o.
nota bene

Выходные данные

Номер подписан в печать: 02-04-2024
Учредитель: Даниленко Василий Иванович, w.danilenko@nbpublish.com
Издатель: ООО <НБ-Медиа>
Главный редактор: Морозов Михаил Николаевич, кандидат технических наук, mikhail.n.morozov@gmail.com
ISSN: 2454-0714
Контактная информация:
Выпускающий редактор - Зубкова Светлана Вадимовна
E-mail: info@nbpublish.com
тел.+7 (966) 020-34-36
Почтовый адрес редакции: 115114, г. Москва, Павелецкая набережная, дом 6А, офис 211.
Библиотека журнала по адресу: http://www.nbpublish.com/library_tariffs.php

Publisher's imprint

Number of signed prints: 02-04-2024
Founder: Danilenko Vasiliy Ivanovich, w.danilenko@nbpublish.com
Publisher: NB-Media ltd
Main editor: Morozov Mikhail Nikolaevich, kandidat tekhnicheskikh nauk, mikhail.n.morozov@gmail.com
ISSN: 2454-0714
Contact:
Managing Editor - Zubkova Svetlana Vadimovna
E-mail: info@nbpublish.com
тел.+7 (966) 020-34-36
Address of the editorial board : 115114, Moscow, Paveletskaya nab., 6A, office 211 .
Library Journal at : http://en.nbpublish.com/library_tariffs.php

Редакционный совет

Гвинианидзе Темур Николаевич - Доктор технических наук, профессор, Государственный университет им. Ак. Церетели Грузия, г. Кутаиси, пр. Тамар-мепе 59. П.и 4600. temuri1951@mail.ru

Council of Editors

Требования к статьям

Журнал является научным. Направляемые в издательство статьи должны соответствовать тематике журнала (с его рубрикатором можно ознакомиться на сайте издательства), а также требованиям, предъявляемым к научным публикациям.

Рекомендуемый объем от 12000 знаков.

Структура статьи должна соответствовать жанру научно-исследовательской работы. В ее содержании должны обязательно присутствовать и иметь четкие смысловые разграничения такие разделы, как: предмет исследования, методы исследования, апелляция к оппонентам, выводы и научная новизна.

Не приветствуется, когда исследователь, трактуя в статье те или иные научные термины, вступает в заочную дискуссию с авторами учебников, учебных пособий или словарей, которые в узких рамках подобных изданий не могут широко излагать свое научное воззрение и заранее оказываются в проигрышном положении. Будет лучше, если для научной полемики Вы обратитесь к текстам монографий или диссертационных работ оппонентов.

Не превращайте научную статью в публицистическую: не наполняйте ее цитатами из газет и популярных журналов, ссылками на высказывания по телевидению.

Ссылки на научные источники из Интернета допустимы и должны быть соответствующим образом оформлены.

Редакция отвергает материалы, напоминающие реферат. Автору нужно не только продемонстрировать хорошее знание обсуждаемого вопроса, работ ученых, исследовавших его прежде, но и привнести своей публикацией определенную научную новизну.

Не принимаются к публикации избранные части из диссертаций, книг, монографий, поскольку стиль изложения подобных материалов не соответствует журнальному жанру, а также не принимаются материалы, публиковавшиеся ранее в других изданиях.

В случае отправки статьи одновременно в разные издания автор обязан известить об этом редакцию. Если он не сделал этого заблаговременно, рискует репутацией: в дальнейшем его материалы не будут приниматься к рассмотрению.

Уличенные в плагиате попадают в «черный список» издательства и не могут рассчитывать на публикацию. Информация о подобных фактах передается в другие издательства, в ВАК и по месту работы, учебы автора.

Статьи представляются в электронном виде только через сайт издательства http://www.e-notabene.ru кнопка "Авторская зона".

Статьи без полной информации об авторе (соавторах) не принимаются к рассмотрению, поэтому автор при регистрации в авторской зоне должен ввести полную и корректную информацию о себе, а при добавлении статьи - о всех своих соавторах.

Не набирайте название статьи прописными (заглавными) буквами, например: «ИСТОРИЯ КУЛЬТУРЫ…» — неправильно, «История культуры…» — правильно.

При добавлении статьи необходимо прикрепить библиографию (минимум 10–15 источников, чем больше, тем лучше).

При добавлении списка использованной литературы, пожалуйста, придерживайтесь следующих стандартов:

- ГОСТ 7.1-2003 Библиографическая запись. Библиографическое описание. Общие требования и правила составления.
- ГОСТ 7.0.5-2008 Библиографическая ссылка. Общие требования и правила составления

В каждой ссылке должен быть указан только один диапазон страниц. В теле статьи ссылка на источник из списка литературы должна быть указана в квадратных скобках, например, [1]. Может быть указана ссылка на источник со страницей, например, [1, с. 57], на группу источников, например, [1, 3], [5-7]. Если идет ссылка на один и тот же источник, то в теле статьи нумерация ссылок должна выглядеть так: [1, с. 35]; [2]; [3]; [1, с. 75-78]; [4]....
А в библиографии они должны отображаться так:
[1]
[2]
[3]
[4]....

Постраничные ссылки и сноски запрещены. Если вы используете сноску, не содержащую ссылку на источник, например, разъяснение термина, включите сноску в текст статьи.

После процедуры регистрации необходимо прикрепить аннотацию на русском языке, которая должна состоять из трех разделов: Предмет исследования; Метод, методология исследования; Новизна исследования, выводы.

Прикрепить 10 ключевых слов.

Прикрепить саму статью.

Требования к оформлению текста:

- Кавычки даются уголками (« ») и только кавычки в кавычках — лапками (“ ”).
- Тире между датами дается короткое (Ctrl и минус) и без отбивок.
- Тире во всех остальных случаях дается длинное (Ctrl, Alt и минус).
- Даты в скобках даются без г.: (1932–1933).
- Даты в тексте даются так: 1920 г., 1920-е гг., 1540–1550-е гг.
- Недопустимо: 60-е гг., двадцатые годы двадцатого столетия, двадцатые годы XX столетия, 20-е годы XX столетия.
- Века, король такой-то и т.п. даются римскими цифрами: XIX в., Генрих IV.
- Инициалы и сокращения даются с пробелом: т. е., т. д., М. Н. Иванов. Неправильно: М.Н. Иванов, М.Н. Иванов.

ВСЕ СТАТЬИ ПУБЛИКУЮТСЯ В АВТОРСКОЙ РЕДАКЦИИ.

По вопросам публикации и финансовым вопросам обращайтесь к администратору
Зубковой Светлане Вадимовне
E-mail: info@nbpublish.com
или по телефону +7 (966) 020-34-36

Подробные требования к написанию аннотаций:

Аннотация в периодическом издании является источником информации о содержании статьи и изложенных в ней результатах исследований.

Аннотация выполняет следующие функции: дает возможность установить основное

содержание документа, определить его релевантность и решить, следует ли обращаться к полному тексту документа; используется в информационных, в том числе автоматизированных, системах для поиска документов и информации.

Аннотация к статье должна быть:

- информативной (не содержать общих слов);
- оригинальной;
- содержательной (отражать основное содержание статьи и результаты исследований);
- структурированной (следовать логике описания результатов в статье);

Аннотация включает следующие аспекты содержания статьи:

- предмет, цель работы;
- метод или методологию проведения работы;
- результаты работы;
- область применения результатов; новизна;
- выводы.

Результаты работы описывают предельно точно и информативно. Приводятся основные теоретические и экспериментальные результаты, фактические данные, обнаруженные взаимосвязи и закономерности. При этом отдается предпочтение новым результатам и данным долгосрочного значения, важным открытиям, выводам, которые опровергают существующие теории, а также данным, которые, по мнению автора, имеют практическое значение.

Выводы могут сопровождаться рекомендациями, оценками, предложениями, гипотезами, описанными в статье.

Сведения, содержащиеся в заглавии статьи, не должны повторяться в тексте аннотации. Следует избегать лишних вводных фраз (например, «автор статьи рассматривает...», «в статье рассматривается…»).

Исторические справки, если они не составляют основное содержание документа, описание ранее опубликованных работ и общеизвестные положения в аннотации не приводятся.

В тексте аннотации следует употреблять синтаксические конструкции, свойственные языку научных и технических документов, избегать сложных грамматических конструкций.

Гонорары за статьи в научных журналах не начисляются.

Цитирование или воспроизведение текста, созданного ChatGPT, в вашей статье

Если вы использовали ChatGPT или другие инструменты искусственного интеллекта в своем исследовании, опишите, как вы использовали этот инструмент, в разделе «Метод» или в аналогичном разделе вашей статьи. Для обзоров литературы или других видов эссе, ответов или рефератов вы можете описать, как вы использовали этот инструмент, во введении. В своем тексте предоставьте prompt - командный вопрос, который вы использовали, а затем любую часть соответствующего текста, который был создан в ответ.

К сожалению, результаты «чата» ChatGPT не могут быть получены другими читателями, и хотя невосстановимые данные или цитаты в статьях APA Style обычно цитируются как личные сообщения, текст, сгенерированный ChatGPT, не является сообщением от человека.

Таким образом, цитирование текста ChatGPT из сеанса чата больше похоже на совместное использование результатов алгоритма; таким образом, сделайте ссылку на автора алгоритма записи в списке литературы и приведите соответствующую цитату в тексте.

Пример:

На вопрос «Является ли деление правого полушария левого полушария реальным или метафорой?» текст, сгенерированный ChatGPT, показал, что, хотя два полушария мозга в некоторой степени специализированы, «обозначение, что люди могут быть охарактеризованы как «левополушарные» или «правополушарные», считается чрезмерным упрощением и популярным мифом» (OpenAI, 2023).

Ссылка в списке литературы

OpenAI. (2023). ChatGPT (версия от 14 марта) [большая языковая модель]. https://chat.openai.com/chat

Вы также можете поместить полный текст длинных ответов от ChatGPT в приложение к своей статье или в дополнительные онлайн-материалы, чтобы читатели имели доступ к точному тексту, который был сгенерирован. Особенно важно задокументировать точный созданный текст, потому что ChatGPT будет генерировать уникальный ответ в каждом сеансе чата, даже если будет предоставлен один и тот же командный вопрос. Если вы создаете приложения или дополнительные материалы, помните, что каждое из них должно быть упомянуто по крайней мере один раз в тексте вашей статьи в стиле APA.

Пример:

При получении дополнительной подсказки «Какое представление является более точным?» в тексте, сгенерированном ChatGPT, указано, что «разные области мозга работают вместе, чтобы поддерживать различные когнитивные процессы» и «функциональная специализация разных областей может меняться в зависимости от опыта и факторов окружающей среды» (OpenAI, 2023; см. Приложение A для полной расшифровки). .

Ссылка в списке литературы

OpenAI. (2023). ChatGPT (версия от 14 марта) [большая языковая модель]. https://chat.openai.com/chat Создание ссылки на ChatGPT или другие модели и программное обеспечение ИИ

Приведенные выше цитаты и ссылки в тексте адаптированы из шаблона ссылок на программное обеспечение в разделе 10.10 Руководства по публикациям (Американская психологическая ассоциация, 2020 г., глава 10). Хотя здесь мы фокусируемся на ChatGPT, поскольку эти рекомендации основаны на шаблоне программного обеспечения, их можно адаптировать для учета использования других больших языковых моделей (например, Bard), алгоритмов и аналогичного программного обеспечения.

Ссылки и цитаты в тексте для ChatGPT форматируются следующим образом:

OpenAI. (2023). ChatGPT (версия от 14 марта) [большая языковая модель]. https://chat.openai.com/chat

Цитата в скобках: (OpenAI, 2023)

Описательная цитата: OpenAI (2023)

Давайте разберем эту ссылку и посмотрим на четыре элемента (автор, дата, название и

источник):

Автор: Автор модели OpenAI.

Дата: Дата — это год версии, которую вы использовали. Следуя шаблону из Раздела 10.10, вам нужно указать только год, а не точную дату. Номер версии предоставляет конкретную информацию о дате, которая может понадобиться читателю.

Заголовок. Название модели — «ChatGPT», поэтому оно служит заголовком и выделено курсивом в ссылке, как показано в шаблоне. Хотя OpenAI маркирует уникальные итерации (например, ChatGPT-3, ChatGPT-4), они используют «ChatGPT» в качестве общего названия модели, а обновления обозначаются номерами версий.

Номер версии указан после названия в круглых скобках. Формат номера версии в справочниках ChatGPT включает дату, поскольку именно так OpenAI маркирует версии. Различные большие языковые модели или программное обеспечение могут использовать различную нумерацию версий; используйте номер версии в формате, предоставленном автором или издателем, который может представлять собой систему нумерации (например, Версия 2.0) или другие методы.

Текст в квадратных скобках используется в ссылках для дополнительных описаний, когда они необходимы, чтобы помочь читателю понять, что цитируется. Ссылки на ряд общих источников, таких как журнальные статьи и книги, не включают описания в квадратных скобках, но часто включают в себя вещи, не входящие в типичную рецензируемую систему. В случае ссылки на ChatGPT укажите дескриптор «Большая языковая модель» в квадратных скобках. OpenAI описывает ChatGPT-4 как «большую мультимодальную модель», поэтому вместо этого может быть предоставлено это описание, если вы используете ChatGPT-4. Для более поздних версий и программного обеспечения или моделей других компаний могут потребоваться другие описания в зависимости от того, как издатели описывают модель. Цель текста в квадратных скобках — кратко описать тип модели вашему читателю.

Источник: если имя издателя и имя автора совпадают, не повторяйте имя издателя в исходном элементе ссылки и переходите непосредственно к URL-адресу. Это относится к ChatGPT. URL-адрес ChatGPT: https://chat.openai.com/chat. Для других моделей или продуктов, для которых вы можете создать ссылку, используйте URL-адрес, который ведет как можно более напрямую к источнику (т. е. к странице, на которой вы можете получить доступ к модели, а не к домашней странице издателя).

Другие вопросы о цитировании ChatGPT

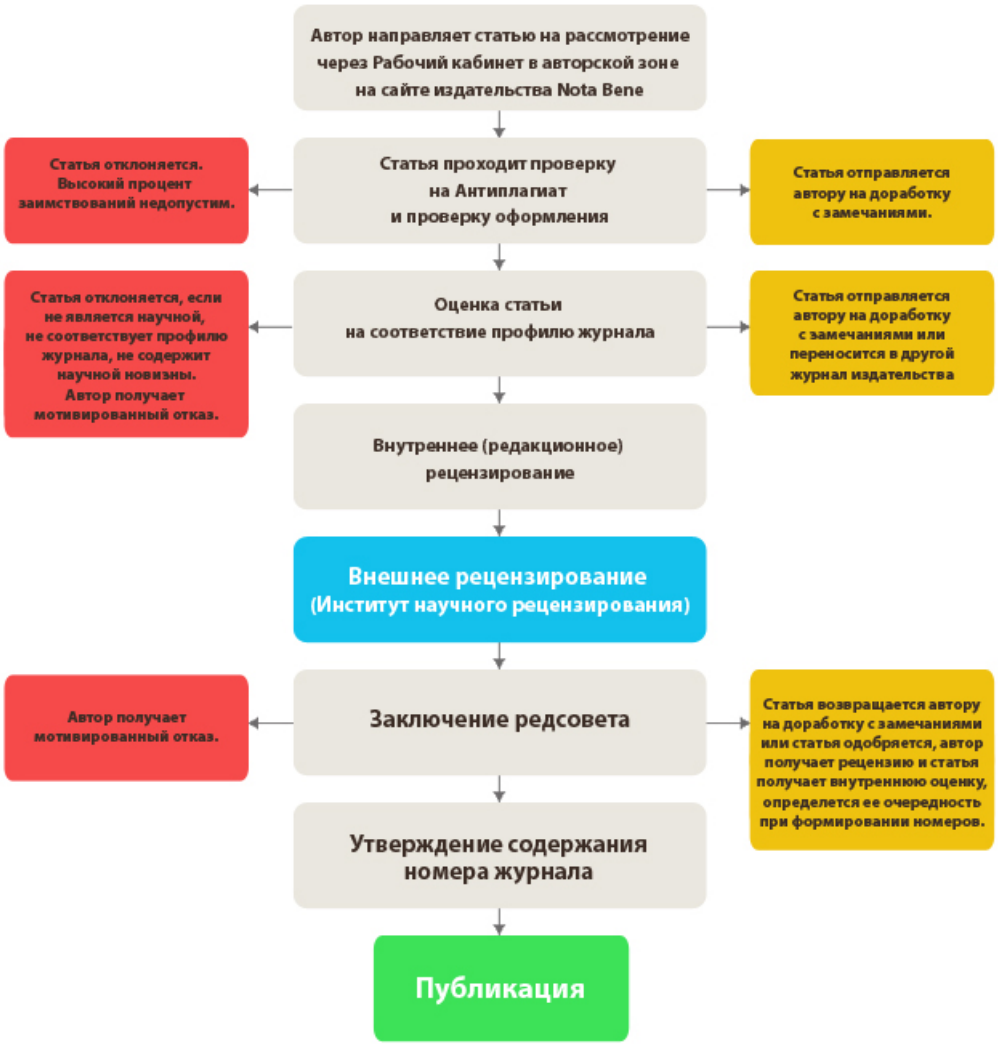
Вы могли заметить, с какой уверенностью ChatGPT описал идеи латерализации мозга и то, как работает мозг, не ссылаясь ни на какие источники. Я попросил список источников, подтверждающих эти утверждения, и ChatGPT предоставил пять ссылок, четыре из которых мне удалось найти в Интернете. Пятая, похоже, не настоящая статья; идентификатор цифрового объекта, указанный для этой ссылки, принадлежит другой статье, и мне не удалось найти ни одной статьи с указанием авторов, даты, названия и сведений об источнике, предоставленных ChatGPT. Авторам, использующим ChatGPT или аналогичные инструменты искусственного интеллекта для исследований, следует подумать о том, чтобы сделать эту проверку первоисточников стандартным процессом. Если источники являются реальными, точными и актуальными, может быть лучше прочитать эти первоисточники, чтобы извлечь уроки из этого исследования, и перефразировать или процитировать эти статьи, если применимо, чем использовать их интерпретацию модели.

Материалы журналов включены:

- в систему Российского индекса научного цитирования;
- отображаются в крупнейшей международной базе данных периодических изданий Ulrich's Periodicals Directory, что гарантирует значительное увеличение цитируемости;
- Всем статьям присваивается уникальный идентификационный номер Международного регистрационного агентства DOI Registration Agency. Мы формируем и присваиваем всем статьям и книгам, в печатном, либо электронном виде, оригинальный цифровой код. Префикс и суффикс, будучи прописанными вместе, образуют определяемый, цитируемый и индексируемый в поисковых системах, цифровой идентификатор объекта — digital object identifier (DOI).

Отправить статью в редакцию

Этапы рассмотрения научной статьи в издательстве NOTA BENE.



Содержание

Черепенин В.А., Смык Н.О., Воробьев С.П. Интеграция облачных, туманных и граничных технологий для оптимизации высоконагруженных систем 1

Алпатов А.Н., Юров И.И. Алгоритм и программная реализация совместного редактирования графических схем в режиме реального времени с использованием библиотеки Socket.IO 10

Малахов С.В., Якупов Д.О., Воробьева Е.Г., Нехаев М.В., Мухтулов М.О., Новосельцева С.В. Развитие и применение операционных систем и оболочек в мобильных технологиях: анализ истории развития и актуальных трендов в сфере мобильных ОС и оболочек 20

Дагаев Д.В. Инструментальный подход к программированию в системе МультиОберон 31

Бондаренко В.А., Попов Д.И. Исследование и разработка алгоритмов к формированию эффективного ансамбля сверточных нейронных сетей для классификации изображений 48

Черепенин В.А., Кацупеев А.А. Анализ подходов к созданию системы «Умная теплица» на основе нейронной сети 68

Англоязычные метаданные 79

Contents

Cherepenin V.A., Smyk N.O., Vorob'ev S.P. Integration of cloud, fog, and edge technologies for the optimization of high-load systems 1

Alpatov A.N., Iurov I.I. Algorithm and software implementation of real-time collaborative editing of graphical schemes using Socket.IO library 10

Malakhov S.V., Yakupov D.O., Vorobeva E.G., Nekhaev M.V., Muhtulov M.O., Novoseltseva S.V. Development and application of operating systems and shells in mobile technologies: analysis of the history of development and current trends in the field of mobile OS and shells 20

Dagaev D.V. Instrumental approach to programming in MultiOberon system 31

Bondarenko V.A., Popov D.I. Research and development of algorithms for the formation of an effective ensemble of convolutional neural networks for image classification 48

Cherepenin V.A., Katsupeev A.A. Analysis of approaches to creating a «Smart Greenhouse» system based on a neural network 68

Metadata in english 79

Программные системы и вычислительные методы
Правильная ссылка на статью:
Черепенин В.А., Смык Н.О., Воробьев С.П. Интеграция облачных, туманных и граничных технологий для оптимизации высоконагруженных систем // Программные системы и вычислительные методы. 2024. № 1. DOI: 10.7256/2454-0714.2024.1.69900 EDN: HYTKBH URL: https://nbpublish.com/library_read_article.php?id=69900

Интеграция облачных, туманных и граничных технологий для оптимизации высоконагруженных систем

Черепенин Валентин Анатольевич

ORCID: 0000-0002-6310-1939

аспирант, кафедра Информационные и измерительные системы и технологии, Южно-Российский государственный политехнический университет (НПИ) имени М.И. Платова

346428, Россия, Ростовская область, г. Новочеркасск, ул. Просвещения, 132

cherept2@gmail.com

Смык Николай Олегович

аспирант, кафедра Программное обеспечение вычислительной техники, Южно-Российский государственный политехнический университет (НПИ) имени М.И. Платова

346428, Россия, Ростовская область, г. Новочеркасск, ул. Просвещения, 132

smyk.n@list.ru

Воробьев Сергей Петрович

кандидат технических наук

доцент, кафедра Информационные и измерительные системы и технологии, Южно-Российский государственный политехнический университет (НПИ) имени М.И. Платова

346428, Россия, Ростовская область, г. Новочеркасск, ул. Просвещения, 132

vsp1999@yandex.ru

Статья из рубрики "Системный анализ, поиск, анализ и фильтрация информации"

DOI:

10.7256/2454-0714.2024.1.69900

EDN:

HYTKBH

Дата направления статьи в редакцию:

13-02-2024

Дата публикации:

20-02-2024

Аннотация: Исследование посвящено анализу методов и инструментов оптимизации работы высоконагруженных систем с использованием облачных, туманных и граничных технологий. Основное внимание уделяется пониманию концепции высоконагруженных систем, выявлению основных причин увеличения нагрузки на такие системы, а также изучению зависимости нагрузки от уровня масштабирования системы, количества пользователей и объема обрабатываемых данных. Введение этих технологий предполагает создание многоуровневой топологической структуры, которая способствует эффективной работе распределенных корпоративных систем и вычислительных сетей. Рассматриваются современные подходы к управлению нагрузками, исследуются основные факторы, влияющие на производительность, и предлагается модель оптимизации, обеспечивающая высокий уровень эффективности и устойчивости системы к пиковым нагрузкам, обеспечивая при этом непрерывность и качество обслуживания конечных пользователей. Методология основана на комплексном подходе, включающем анализ существующих проблем и предложение новаторских решений для оптимизации, применение архитектурных решений на базе IoT, облачных, туманных и граничных вычислений для улучшения производительности и снижения задержек в высоконагруженных системах. Научная новизна данной работы заключается в разработке уникальной многоуровневой топологической структуры, способной интегрировать облачные, туманные и граничные вычисления для оптимизации высоконагруженных систем. Эта структура позволяет обеспечить улучшенную производительность, снижение задержек и эффективное масштабирование системы, решая при этом проблемы управления большими объемами данных и одновременным обслуживанием множества запросов. Выводы исследования подчеркивают значительный потенциал технологии IoT в улучшении производственных процессов, демонстрируя, как интеграция современных технологических решений может способствовать повышению урожайности, качества продукции и управлению рисками. Результаты работы предоставляют основу для дальнейшего развития умного сельского хозяйства и могут быть применены в различных отраслях для создания эффективных, масштабируемых и унифицированных систем, обеспечивая тем самым новые возможности для устойчивого развития аграрного сектора и других сфер экономики.

Ключевые слова:

Высоконагруженные системы, Облачные вычисления, Туманные вычисления, Граничные вычисления, Оптимизация производительности, Масштабируемость, Интернет вещей, Интеграция технологий, Управление данными, Непрерывность сервиса

Введение

Термин "высоконагруженная система" обычно применяется к веб-сервисам и сайтам, испытывающим интенсивное взаимодействие с большим количеством пользователей одновременно. Однако это понятие также распространяется на разнообразные информационные системы и приложения для бизнеса, работающие с значительными объемами данных. Поэтому задачи по оптимизации высоконагруженных систем актуальны не только для веб-разработки, но и в контексте любых проектов, предполагающих наличие серверной и клиентской частей. Следовательно, высоконагруженной системой

считается приложение, испытывающее высокую нагрузку из-за множества пользователей, обширных массивов данных или интенсивности вычислений. Эти аспекты могут проявляться как вместе, так и по отдельности, но наличие хотя бы одного из них указывает на повышенные требования к ресурсам системы.

В контексте развития современных дистрибутивных информационных систем, проектирование включает в себя использование преимуществ технологий Интернета вещей (IoT) и применение архитектурных решений, основанных на облачных, туманных и граничных вычислениях. Облачные вычисления представляют собой метод построения распределенной системы с доступом к гибкому и масштабируемому набору ресурсов через интернет. Туманные вычисления функционируют как промежуточный слой между конечными устройствами и центрами данных, сокращая задержки и сетевой трафик. Граничные вычисления дополнительно приближают обработку данных к пользователю, выполняя большинство операций на периферии сети, что способствует мгновенному реагированию на полученные данные благодаря использованию программируемых логических контроллеров для управления процессами [1].

Разработка эффективной архитектуры для такого сложного взаимодействующего комплекса предполагает использование многоуровневого подхода к представлению топологии вычислительной сети, обеспечивая оптимизацию работы системы в целом.

Методология исследования

Задачей данной статьи является детальное рассмотрение стратегий и инструментария для повышения эффективности функционирования систем с высоким уровнем нагрузки. Исследуемый объект — это системы, обеспечивающие функциональность веб-сайтов, одновременно обрабатывающие запросы множества пользователей. В качестве методологической базы применяются процедуры анализа и синтеза, а также техники обобщения существующих данных и исследований в данной области [2].

Результаты исследования и их обсуждение

Анализируя высоконагруженные системы, ключевое внимание уделяется их специфическим атрибутам:

1. Высоконагруженные системы характеризуются строгой архитектурой с ограниченным пространством для модификаций в отдельных подсистемах. Их сложная внутренняя структура ограничивает возможности для глубокой адаптации, делая систему менее гибкой из-за уникальности каждой конфигурации. Обработка данных в таких системах, требующая высокой стабильности и надежности, предполагает тщательный подбор структур баз данных, исходя из их специфики, объема данных и интенсивности запросов. Попытки увеличить гибкость системы могут привести к значительным затратам ресурсов, поскольку это требует комплексного переосмысления и возможной реорганизации основных принципов ее работы [3].

2. Одним из ключевых атрибутов высоконагруженных систем является их способность обеспечивать мгновенную реакцию. В контексте обработки данных через запросы, быстродействие системы критически важно: задержки в обработке запросов напрямую влияют на время ожидания пользователем необходимой информации.

3. Масштабируемость является ключевым аспектом для систем с высоким уровнем загрузки, поскольку увеличение объема обрабатываемых данных может значительно повысить нагрузку на информационную инфраструктуру. Для адаптации к растущим

требованиям, масштабируемость обычно достигается двумя основными методами. Первый — вертикальное масштабирование, предполагает повышение производительности отдельных элементов системы для усиления её общей мощности. Этот метод не требует структурных изменений в архитектуре и часто реализуется за счёт модернизации оборудования. Второй метод — горизонтальное масштабирование, включает в себя распределение нагрузки между несколькими серверами или узлами, работающими параллельно, что требует добавления новых компонентов в систему и соответствующей настройки программного обеспечения для эффективного взаимодействия между ними. Хотя горизонтальное масштабирование предполагает большую сложность в реализации, оно обеспечивает более гибкое и масштабируемое решение в долгосрочной перспективе и часто выбирается как наиболее предпочтительный вариант [4]. Важно тщательно анализировать потребности системы и определять оптимальный подход к масштабированию, учитывая специфику и требования к производительности.

4. Разработка модульной архитектуры для высоконагруженных систем предусматривает организацию их структуры в виде отдельных компонентов, которые затем интегрируются на разнообразные серверные платформы. Этот подход позволяет распределить часть нагрузки, назначая особо интенсивно используемые модули на множество серверов для их параллельной работы. Хотя такое решение улучшает производительность, оно может привести к проблемам с согласованностью данных из-за их параллельной обработки [5]. С увеличением общей нагрузки на систему усиливается риск возникновения несогласованности данных, вызванной одновременными операциями разных пользователей. Поэтому предпочтительнее выбирать стратегию, при которой интенсивно используемые процессы выделяются на отдельные, более мощные сервера без параллелизации, что способствует повышению общей эффективности и надежности системы.

5. Интенсивная нагрузка на интеграционный уровень системы напрямую связана с её модульной структурой. Расширение системы за счёт добавления новых модулей увеличивает количество взаимодействий между ними, требуя от коммуникационных процессов высокой скорости и надёжности. С увеличением числа модулей сложность этих взаимодействий возрастает, что ведёт к усиленной загрузке на интеграционный слой системы, особенно когда объём коммуникаций увеличивается экспоненциально.

6. Уникальность является ключевым атрибутом высоконагруженных приложений, подчёркивая отсутствие унифицированных стандартных подходов к их разработке. Это означает, что каждое решение разрабатывается с учётом специфических потребностей и требований конкретного бизнеса, делая систему неповторимой и точно адаптированной под задачи заказчика [6].

7. Эффективное применение стратегии редундантности для ключевых элементов системы становится критичным для поддержания непрерывности бизнес-процессов, особенно в контексте высоконагруженных систем. Для гарантии стабильности работы такие системы оборудуются дублирующими узлами, как в программном, так и в аппаратном аспектах. Это не подразумевает постоянную параллельную активность всех резервных компонентов, а скорее предусматривает их готовность к моментальному включению в работу при возникновении критических нагрузок, обеспечивая таким образом разгрузку основной системы и её бесперебойное функционирование.

Высокие нагрузки часто определяются не самой архитектурой системы, а условиями её эксплуатации, отражая специфику деятельности в определённой бизнес-сфере. Среди

причин, влияющих на увеличение нагрузки, можно выделить:

- Увеличение числа запросов к системам управления взаимоотношениями с клиентами (CRM) и планирования ресурсов предприятия (ERP), обслуживающих множество пользователей, а также в поддержке клиентов и контакт-центрах, где происходит обработка значительного количества звонков и обращений;

- Рост объёма обрабатываемой информации в системах мониторинга с большим количеством подключённого оборудования, в инструментах бизнес-аналитики, а также в CRM и ERP системах, обрабатывающих обширные массивы данных;

- Ошибки в настройке системы, часто возникающие из-за недочётов в программном коде или отсутствия должной оптимизации, приводящие к усилению нагрузки на серверные ресурсы.

Для снижения и управления нагрузкой на систему применяются различные стратегии оптимизации, включающие:

- Применение сетевых протоколов и внешних библиотек для уменьшения числа запросов, включая методы кэширования данных, как на уровне запросов к базе данных, так и при получении ответов от сервера, что способствует снижению задержек и увеличению производительности системы;

- Оптимизация работы с базой данных через индексирование, что ускоряет поиск и обработку данных, репликацию для распределения нагрузки и обеспечения отказоустойчивости, а также секционирование для эффективного управления и хранения больших объемов информации [7].

Детализируя методы оптимизации взаимодействия с базой данных, следует выделить несколько ключевых направлений:

1. Сериализация и десериализация данных играют важную роль в процессе обмена информацией между сервером и клиентом. Эти процедуры обеспечивают корректную передачу данных, но также влекут за собой временные затраты. Оптимизация данных процессов позволяет ускорить обработку информации и улучшить общую производительность системы.

2. Кэширование запросов к базе данных становится эффективным решением для снижения нагрузки на информационные хранилища. Организация кэша позволяет временно сохранять результаты часто выполняемых или редко изменяемых запросов, существенно уменьшая количество обращений к базе данных. Для реализации кэширования часто применяются хэш-таблицы, обеспечивающие быстрый доступ к сохраненным данным.

3. Индексирование базы данных представляет собой создание специализированных структур (индексов), которые способствуют ускорению поиска данных. Индексы не только повышают скорость доступа к информации, но и поддерживают целостность данных. Однако, следует учитывать, что операции с индексами, такие как их перестроение после удаления данных, могут потребовать дополнительных ресурсов. Применение индексов оправдано в базах данных с большим объемом записей, где они могут значительно оптимизировать процессы поиска и обработки информации.

Эти методы оптимизации позволяют улучшить производительность высоконагруженных систем, снизить время ответа на запросы и увеличить общую эффективность работы с

данными [8].

Оптимизация доступа к данным в базах данных может включать стратегию репликации, которая служит для увеличения пропускной способности и доступности системы. Репликация достигается путем создания копий информационной базы, разделяя роли между первичными (ведущими) и вторичными (ведомыми) узлами для балансировки нагрузки операций чтения и обеспечения актуальности данных через синхронизацию. Ведущий узел обрабатывает как чтение, так и запись, распространяя обновления по ведомым узлам. Существуют различные методы репликации:

1. Синхронная репликация обеспечивает полную согласованность данных между узлами, требуя подтверждения записи от всех участников, что повышает надежность, но увеличивает время отклика.

2. Асинхронная репликация ускоряет работу системы, переключаясь на следующие операции сразу после записи на ведущем узле без ожидания подтверждения от ведомых, чем жертвует гарантией актуальности данных на всех узлах.

3. Модель "Ведущий-Ведомые" предполагает наличие одного ведущего узла, обрабатывающего все операции, и множества ведомых, синхронизирующихся через фиксированные интервалы. Это упрощает структуру, но создает риски при отказе ведущего узла.

4. Модель "Ведущие-Ведомые" вносит дополнительную гибкость, добавляя несколько ведущих узлов и повышая отказоустойчивость за счет усложнения процесса синхронизации данных между ними.

Каждый из этих подходов имеет свои преимущества и недостатки, выбор метода репликации зависит от специфических требований к производительности, доступности и консистентности данных в конкретной системе.

Оптимизация баз данных через секционирование представляет собой стратегию разделения данных на меньшие сегменты, направленную на улучшение производительности и пропускной способности [9]. Этот подход позволяет балансировать нагрузку, предотвращая перегрузку отдельных узлов и способствуя более эффективной обработке данных. Секционирование может осуществляться разными методами:

- По диапазону ключей – это базовый метод, при котором данные распределяются согласно определенным ключам. Главным вызовом здесь является выбор ключа, который обеспечит оптимальное разделение данных.

- С использованием хэш-функций – более продвинутая техника, где специальная хэш-функция равномерно распределяет данные по секциям. Этот метод обеспечивает более сбалансированное распределение данных и эффективность работы системы.

Для поддержания оптимальной производительности системы после секционирования необходимо регулярно проводить ребалансировку [10]. Это связано с тем, что запись данных в разные секции может происходить неравномерно, что со временем может привести к замедлению скорости обработки запросов.

Заключение

В завершении обсуждения стратегий оптимизации важно подчеркнуть, что эффективное повышение производительности системы требует комплексного подхода, включающего

применение множества разнообразных методик. Каждая из них обладает своими уникальными особенностями и предполагает различные уровни воздействия на архитектуру и процессы взаимодействия в системе. Выбор конкретного метода или их комбинации должен базироваться на тщательном анализе текущего состояния системы, определении ключевых точек, требующих оптимизации, а также на учете специфических потребностей и ожиданий заказчика.

Среди возможных направлений для улучшения производительности могут быть выделены:

- Анализ и оптимизация алгоритмов работы с данными, включая пересмотр запросов к базам данных, использование более эффективных алгоритмов сериализации и десериализации данных.

- Кэширование часто запрашиваемой информации для снижения нагрузки на базы данных и ускорения доступа к данным.

- Масштабирование системы, как вертикальное, так и горизонтальное, для обеспечения гибкости и масштабируемости в ответ на растущие нагрузки.

- Репликация и секционирование баз данных для улучшения доступности и распределения нагрузки между серверами.

- Использование современных подходов к разработке, включая микросервисную архитектуру, для повышения модульности и упрощения масштабирования и обновления системы.

Таким образом, стратегия оптимизации должна быть многоаспектной и учитывать как технические возможности существующей инфраструктуры, так и долгосрочные бизнес-цели заказчика. Разработчикам необходимо гибко подходить к выбору инструментов и методик, адаптируя их под конкретные задачи и условия эксплуатации системы, чтобы достичь оптимального соотношения между производительностью, стабильностью и расширяемостью.

Библиография

1. Catal, C.; Tekinerdogan, B. Aligning education for the life sciences domain to support digitalization and Industry 4.0. Procedia Computer Science. 2019, 158, 99–106. DOI:10.1016/j.procs.2019.09.032
2. Patel, C.; Doshi, N. A novel MQTT security framework in generic IoT model. Procedia Computer Science. 2020, 171, 1399–1408. DOI:10.1016/j.procs.2020.04.150
3. Subeesh, A.; Mehta, C.R. Automation and digitization of agriculture using artificial intelligence and internet of things. Artificial Intelligence in Agriculture. 2021, 5, 278–291. DOI:10.1016/j.aiia.2021.11.004
4. Faridi, F.; Sarwar, H.; Ahtisham, M.; Kumar, S.; Jamal, K. Cloud computing approaches in health care. Materials Today: Proceedings, 2022, 51, 1217–1223. DOI:10.1016/j.matpr.2021.07.210
5. Tzounis, A.; Katsoulas, N.; Bartzanas, T.; Kittas, C. Internet of Things in agriculture, recent advances and future challenges. Biosystems Engineering. 2017, 164, 31–48. DOI:10.1016/j.biosystemseng.2017.09.007
6. Tao, W.; Zhao, L.; Wang, G.; Liang, R. Review of the internet of things communication technologies in smart agriculture and challenges. Computers and Electronics in Agriculture. 2021, 189, 106352. DOI:10.1016/j.compag.2021.106352

7. Moysiadis, V.; Sarigiannidis, P.; Vitsas, V.; Khelifi, A. Smart farming in Europe. Computer Science Review. 2021, 39, 100345. DOI:10.1016/j.cosrev.2020.100345
8. Raj, M.; Gupta, S.; Chamola, V.; Elhence, A.; Garg, T.; Atiquzzaman, M.; Niyato, D. A survey on the role of Internet of Things for adopting and promoting Agriculture 4.0. Journal of Network and Computer Applications. 2021, 187, 103107. DOI:10.1016/j.jnca.2021.103107
9. Boursianis, A.D.; Papadopoulou, M.S.; Diamantoulakis, P.; Liopa-Tsakalidi, A.; Barouchas, P.; Salahas, G.; Karagiannidis, G.; Wan, S.; Goudos, S.K. Internet of Things (IoT) and agricultural unmanned Aerial Vehicles (UAVs) in smart farming: A comprehensive review. Internet of Things. 2022, 18, 100187. DOI:10.1016/j.iot.2020.100187
10. Singh, S.; Chana, I.; Buyya, R. Agri-Info: Cloud based autonomic system for delivering agriculture as a service. Internet of Things. 2020, 9, 100131. DOI:10.1016/j.iot.2019.10013

Результаты процедуры рецензирования статьи

В связи с политикой двойного слепого рецензирования личность рецензента не раскрывается.
Со списком рецензентов издательства можно ознакомиться здесь.

Рецензируемая работа посвящена исследованию интеграции облачных, туманных и граничных технологий для оптимизации высоконагруженных систем.
Методология исследования базируется на изучении и обобщении научных публикаций по рассматриваемой теме, применении процедур анализа и синтеза, а также техники обобщения существующих данных и исследований в данной области.
Актуальность работы авторы связывают с тем, что задачи оптимизации высоконагруженных систем актуальны не только для веб-разработки, но и в контексте любых проектов, предполагающих наличие серверной и клиентской частей.
Научная новизна рецензируемого исследования, по мнению рецензента, состоит в обобщении стратегий и инструментария для повышения эффективности функционирования систем с высоким уровнем нагрузки.
В тексте статьи выделены следующие разделы: Введение, Методология исследования, Результаты исследования и их обсуждение, Заключение, Библиография.
В статье высоконагруженной системой считается приложение, испытывающее высокую нагрузку из-за множества пользователей, обширных массивов данных или интенсивности вычислений. Особое внимание авторы уделяют специфическим атрибутам высоконагруженных систем: строгой архитектуре с ограниченным пространством для модификаций в отдельных подсистемах; способности обеспечивать мгновенную реакцию; масштабируемости; организация их структуры в виде отдельных компонентов, которые затем интегрируются на разнообразные серверные платформы; модульная структура, уникальность, применение стратегии редундантности для ключевых элементов системы.
В публикации выделены причины, влияющие на увеличение нагрузки системы; названы применяемые для управления нагрузкой стратегии оптимизации; выделены ключевые направлений методов оптимизации, которые позволяют улучшить производительность высоконагруженных систем, снизить время ответа на запросы и увеличить общую эффективность работы с данными; рассмотрены подходы к репликации в зависимости от специфических требований к производительности, доступности и консистентности данных в конкретной системе. В Заключении отмечено, что эффективное повышение производительности системы требует комплексного подхода, отражены возможные

направления для улучшения производительности, достижения оптимального соотношения между производительностью, стабильностью и расширяемостью системы.

Библиографический список включает 10 источников – научные публикации на английском языке по рассматриваемой теме, на которые в тексте приведены адресные ссылки, что подтверждает наличие апелляции к оппонентам.

В качестве замечаний, следует отметить отсутствие ссылок на научные работы, опубликованные на русском языке ¬– представляется, что существуют и русскоязычные публикации, заслуживающие внимания; в тексте встречаются несогласованные предложения, например, «Анализируя высоконагруженные системы, ключевое внимание уделяется их специфическим атрибутам…» и др.

Рецензируемый материал соответствует направлению журнала «Программные системы и вычислительные методы», отражает результаты проведенной авторами работы, может вызвать интерес у читателей, поскольку содержит интересные сведения об интеграции облачных, туманных и граничных технологий для оптимизации высоконагруженных систем.

Программные системы и вычислительные методы
Правильная ссылка на статью:
Алпатов А.Н., Юров И.И. Алгоритм и программная реализация совместного редактирования графических схем в режиме реального времени с использованием библиотеки Socket.IO // Программные системы и вычислительные методы. 2024. № 1. DOI: 10.7256/2454-0714.2024.1.70173 EDN: PQMMUM URL: https://nbpublish.com/library_read_article.php?id=70173

Алгоритм и программная реализация совместного редактирования графических схем в режиме реального времени с использованием библиотеки Socket.IO

Алпатов Алексей Николаевич

ORCID: 0000-0001-8624-1662

кандидат технических наук

доцент, кафедра инструментального и прикладного программного обеспечения, Федеральное государственное бюджетное образовательное учреждение высшего образования «МИРЭА — Российский технологический университет»

119454, Россия, Московская область, г. Москва, проспект Вернадского, 78, каб. Г-225

alpatov@mirea.ru

Юров Илья Игоревич

ORCID: 0009-0003-7121-4321

магистр, кафедра инструментального и прикладного программного обеспечения, Федеральное государственное бюджетное образовательное учреждение высшего образования «МИРЭА — Российский технологический университет»

109383, Россия, Московская область, г. Москва, ул. Полбина, 35к2, 912

frit_027@mail.ru

Статья из рубрики "Языки программирования"

DOI:

10.7256/2454-0714.2024.1.70173

EDN:

PQMMUM

Дата направления статьи в редакцию:

20-03-2024

Дата публикации:

31-03-2024

Аннотация: В современном мире командная работа становится все более распространенной. Разные участники могут находиться в разных местах, но им все равно нужно работать вместе над одним проектом, в том числе и над графическими схемами. Важным аспектом такого подхода является возможность наблюдать изменения, вносимые другими участниками, в режиме реального времени. Это позволяет, прежде всего, снизить частоту конфликтов при одновременном редактировании одного и того же элемента схемы. Однако существующие решения для обмена данными при совместном редактировании графических схем в режиме реального времени сталкиваются с рядом проблем, такими как задержки при передаче данных. Предметом исследования в настоящей статье является разработка минимально жизнеспособного веб-приложения, позволяющего пользователям осуществлять совместное графическое редактирование полотна в режиме реального времени. Объектом исследования выступает модель процесса совместного редактирования в реальном времени с учетом разрешения возникающих конфликтов. Методология исследования основана на теоретическом подходе по выявлению математических формул, описывающих изменение состояния документа при его совместном редактировании пользователями. Дана характеристика применения протоколов HTTP и WebSocket в многопользовательских клиент-серверных приложениях. Для применения протокола WebSocket используется библиотека Socket.IO. Сервер приложения построен с помощью фреймворка Express. Основным вкладом авторов в исследование темы является модель процесса совместного редактирования в реальном времени, а также механизм определения конфликтов для любого количества пользователей и функция разрешения конфликтов для каждой пары конфликтующих изменений при совместном редактировании документов в режиме онлайн. В рамках данного исследования дополнительно предложен алгоритм совместного редактирования графических схем в режиме реального времени и дана его реализация в виде программной системы. Предложенный в результате исследования алгоритм на языке программирования JavaScript может быть использован в качестве основы для разработки более многофункциональных веб-приложений с использованием библиотеки Socket.IO и являться объектом будущих исследований, затрагивающих многопользовательское взаимодействие и разрешение конфликтов в режиме реального времени.

Ключевые слова:

протокол HTTP, протокол WebSocket, клиент-серверное приложение, совместное редактирование, определение конфликтов, разрешение конфликтов, язык программирования JavaScript, графическая схема, алгоритм, управляемость событиями

Введение

Протокол HTTP появился в конце XX века и за это время стал преобладающей технологией передачи данных, применяемой при разработке клиент-серверных приложений в веб-среде [1]. В основе работы протокола HTTP лежит принцип «запрос-ответ», когда клиент посылает нужный запрос, например, GET или POST, а сервер в зависимости от типа запроса выполняет необходимые действия и возвращает клиенту нужные данные. Для реализации алгоритма совместного редактирования графических схем в режиме реального времени в протоколе HTTP выявляются существенные недостатки: отсутствие двунаправленной и полнодуплексной связи между компонентами

системы [2]. Вследствие этого процесс обмена координатами курсора мыши между сервером и клиентами при рисовании линий в браузере может вызвать задержки и оказывать дополнительную нагрузку на сервер.

Обозначенных недостатков лишен протокол WebSocket. Технология позволяет веб-серверу и браузеру обмениваться данными по одному и тому же соединению одновременно в обоих направлениях [3]. Сервер в любой момент времени может отправить сообщение сразу всем клиентам, ранее установившим с ним соединение. Точно также любой клиент по тому же соединению может отправить нужные данные на сервер [4].

Но при использовании протокола WebSocket без вспомогательных библиотек реализация алгоритма совместного редактирования графических схем в режиме реального времени потребует разработки дополнительных процедур для обеспечения бесперебойной и стабильной работы компонентов системы. В рамках данной работы будет предложено решение данной задачи за счет использования JavaScript-библиотеки Socket.IO, которая добавляет управляемость событиями, автоматическое переподключение при разрыве соединения, мультиплексирование и другие функции [5].

Модель процесса совместного редактирования в реальном времени

Пусть есть два пользователя, A и B, которые работают над одной графической схемой в разных процессах. Пусть D(t) представляет текущее состояние документа в момент времени t в процессе совместного редактирования графической схемы. Предположим, что каждый пользователь вносит изменения в документ независимо от других пользователей, и все изменения синхронизируются в реальном времени.

Тогда процесс совместного редактирования можно описать следующей формулой:

$$D(t) = D_0 + \Delta D_1(t) + \Delta D_2(t) + \cdots + \Delta D_n(t), \quad (1)$$

где D_0 — начальное состояние документа до начала редактирования, $\Delta D_i(t)$ — изменения, внесенные i-ым пользователем в момент времени t, n — количество пользователей, участвующих в редактировании.

Таким образом, текущее состояние документа D(t) представляет собой сумму начального состояния документа и всех изменений, внесенных каждым пользователем в реальном времени. Такой подход отражает динамику процесса совместного редактирования графической схемы в веб-приложениях и учитывает вклад каждого участника в формирование итогового состояния документа.

В ходе совместного редактирования документов в режиме онлайн могут возникать различные конфликты (конфликт вставки, удаления, форматирования, перемещения и т. д.) [6]. Рассмотрим процесс возникновения конфликтов данных в веб-приложении для совместного редактирования документов. Пусть каждый пользователь, редактирующий графическую схему, имеет свой локальный набор данных о состоянии документа. Пусть $D_A(t)$ и $D_B(t)$ представляют текущее состояние документа для пользователя A и пользователя B в момент времени t. Тогда можно определить конфликт следующим образом: конфликт возникает, если пользователя A и B внесли изменения в один и тот же узел или связь графической схемы в период времени Δt — интервал времени, в течение которого изменения могут считаться конфликтными.

Иными словами, оба пользователя начинают редактировать документ в момент времени t_0. Пользователь A внес изменения в документ, приводя к изменению его состояния от $D(t_0)$ до $D_A(t_1)$ в момент времени t_1, или $D_A(t_1) = D(t_0) + \Delta D_A(t_1)$. Пользователь B также внес изменения в документ, что приводит к изменению его состояния от $D(t_0)$ до $D_B(t_2)$ в момент времени t_2 или $D_B(t_1) = D(t_0) + \Delta D_B(t_2)$. Теперь если $t_1 < t_2$, то возникает конфликт, так как оба пользователя внесли изменения в документ в разное время.

Формула определения конфликтов времени редактирования может быть записана следующим образом:

$$\Psi(D_A(t), D_B(t), \Delta_t) = \{(n, t) \mid n, t \in [t, t + \Delta t]\}, \quad (2)$$

где n — узел или связь, (n, t) представляет изменение, внесенное пользователем в узел или связь n в момент времени t.

Формула определения конфликтов для любого количества пользователей n может быть записана следующим образом:

$$\Psi\left(D_{U_1}(t), D_{U_2}(t), \ldots D_{U_n}(t), \Delta_t\right) = \{(n, t) \mid n, t \in [t, t + \Delta t]\}, \quad (3)$$

где n — узел или связь.

Если для пользователей A и B обнаружены одинаковые изменения в одном и том же узле или связи в течение интервала Δt, то это считается конфликтом редактирования.

Пусть

$$D_{U_i}(t) = \left\{\delta_{i,1}(t), \delta_{i,2}(t), \ldots, \delta_{i,k_i}(t)\right\} \quad (4)$$

— это набор изменений, внесенныхпользователем U_i в момент времени t, где k_i — количество изменений.

Для каждого изменения $\delta_{i,j}(t)$ определим его совместимость с текущим состоянием документа как $C_{i,j} = f(\delta_{i,j}(t), D(t))$. Тогда конфликт совместимости изменений (два изменения несовместимы) определяется через

$$\Psi_{i,j,k,l}(t) \begin{cases} 1, \text{если } C_{i,j}(t) \neq C_{k,l}(t) \\ 0, \text{иначе} \end{cases}. \quad (5)$$

Тогда для каждой пары конфликтующих изменений можно применить функцию разрешения конфликтов вида

$$\Theta_{i,j,k,l}(t) = g\left(\delta_{i,j}(t), \delta_{k,l}(t), D(t)\right), \quad (6)$$

где $\Theta_{i,j,k,l}(t)$ — результат разрешения конфликта между изменениями $\delta_{i,j}(t)$ и $\delta_{k,l}(t)$ в момент времени t, $\delta_{i,j}(t)$ и $\delta_{k,l}(t)$ — изменения, которые конфликтуют между собой, D(t) представляет текущее состояние документа в момент времени t.

Функция g применяется для разрешения конфликта между этими двумя изменениями. Она принимает два конфликтующих изменения $\delta_{i,j}(t)$ и $\delta_{k,l}(t)$ а также текущее состояние документа D(t) и возвращает разрешенное изменение. Ее задача — определить, какое из этих изменений следует применить к документу. Функция g может быть реализована с

использованием различных алгоритмов и стратегий разрешения конфликтов и в зависимости от характера конфликта. Например, если изменения $\delta_{i,j}(t)$ и $\delta_{k,l}(t)$ не взаимоисключающие, функция g может объединить их в одно изменение. Практически это может быть использовано для разрешения ситуации, когда одно изменение вставляет текст, а другое изменение добавляет форматирование этого текста, их можно объединить в одно изменение, которое вставляет текст с форматированием. Так $\delta_1(t)$ представляет изменение, вставляющее текст $Text_1$ в позицию Pos_1, а $\delta_2(t)$ — изменение, добавляющее форматирование $Format_2$ к этому тексту. Тогда функция g, примет следующий вид:

$$g\left(\delta_{i,j}(t), \delta_{k,l}(t), D(t)\right) = \varphi_{\text{merged}}(t), \quad (7)$$

где $\varphi_{merged}(t)$ — объединенное изменение, которое вставляет текст $Text_1$ с применением форматирования $Format_2$ в позицию Pos_1.

Алгоритм совместного редактирования графических схем в режиме реального времени

Реализацию алгоритма следует начать с написания серверной части приложения. Сервер с нужным портом можно создать различными способами, но в данном исследовании использовался минималистичный фреймворк Express [7]. После этого следует инициализировать экземпляр io с помощью класса Server, предоставляемый библиотекой Socket.IO, передав в его конструктор ранее созданный объект сервера [8]. Получив WebSocket-сервер, необходимо добавить слушателей событий и обработчики для них. В данном алгоритме сервер ожидает два события: connection и draw. При возникновении первого события сервер генерирует событие color и передает вновь подключившемуся клиенту случайный HEX-код цвета для графических линий пользователя. При возникновении события draw сервер отправляет полученные координаты точки перемещения курсора мыши конкретного пользователями всем остальным клиентам системы по тому же соединению. На рисунке 1 представлен код описанной части алгоритма с экземпляром io.

```
37  const io :Server<DefaultEventsMap, DefaultEventsMap, DefaultEventsMap, any>  = new Server(server);
38
39  io.on( ev: 'connection',  listener: (socket :Socket<DefaultEventsMap, DefaultEventsMap, DefaultEventsMap, any> ) :void  => {
40      socket.emit( ev: 'color',  args: { color: getColor() });
41
42      socket.on( ev: 'draw',  listener: (data) :void  => {
43          socket.broadcast.emit( ev: 'draw', data);
44      });
45  });
```

Рисунок 1. Серверная часть алгоритма с экземпляром io

В клиентской части алгоритма в первую очередь следует инициализировать экземпляр socket и подписать его на два события: color и draw. В обработчике первого события в переменную strokeStyle будет сохраняться цвет, полученный с сервера. В свою очередь, в обработчике draw клиент будет получать с сервера координаты курсора мыши другого пользователя и строить по ним графическую линию на своем экране с помощью специальной функции drawLine(). Стоит отметить, что сама линия отображается с помощью стандартных средств языка JavaScript, а именно, инструмента Canvas API [9].

Затем нужно проинициализировать необходимые переменные начальными значениями, получить элемент canvas из HTML-кода страницы и привязать к нему три события:

mousedown (пользователь нажал мышью по графической области), mousemove (пользователь двигает мышью с зажатой кнопкой) и mouseup (пользователь опустил кнопку мыши) [10]. У каждого из указанных событий есть свой обработчик. При возникновении события mousedown алгоритм сохраняет начальные координаты startX и startY курсора мыши в созданный ранее объект mouse и устанавливает переменную isDraw в значение true. При событии mousemove, если значение isDraw истинно, сохраняются следующие координаты endX и endY курсора мыши, после чего все данные отправляются на сервер по соединению веб-сокета события draw и строится графическая линия текущего пользователя.

Затем в свойства объекта mouse startX и startY записываются значения endX и endY соответственно. На рисунке 2 продемонстрирован код обработчика события mousemove.

```
34  canvas.addEventListener( type: 'mousemove', listener: (e : MouseEvent ) : void => {
35      if (isDraw) {
36          const rect : DOMRect = e.target.getBoundingClientRect();
37          mouse.endX = e.pageX - rect.left;
38          mouse.endY = e.pageY - rect.top;
39
40          const coordinates : {…} = {
41              startX: mouse.startX,
42              startY: mouse.startY,
43              endX: mouse.endX,
44              endY: mouse.endY,
45          };
46
47          const data : {…} = { ...coordinates, strokeStyle }
48          socket.emit( ev: 'draw', data);
49          drawLine(context, data);
50
51          mouse.startX = mouse.endX;
52          mouse.startY = mouse.endY;
53      }
54  });
```

Рисунок 2. Код обработчика события mousemove

При возникновении последнего события mouseup переменная isDraw устанавливается в значение false.

После локального запуска сервера можно проверить работу алгоритма, нарисовав несколько простых геометрических фигур в трех разных браузерах. Результат показан на рисунке 3.

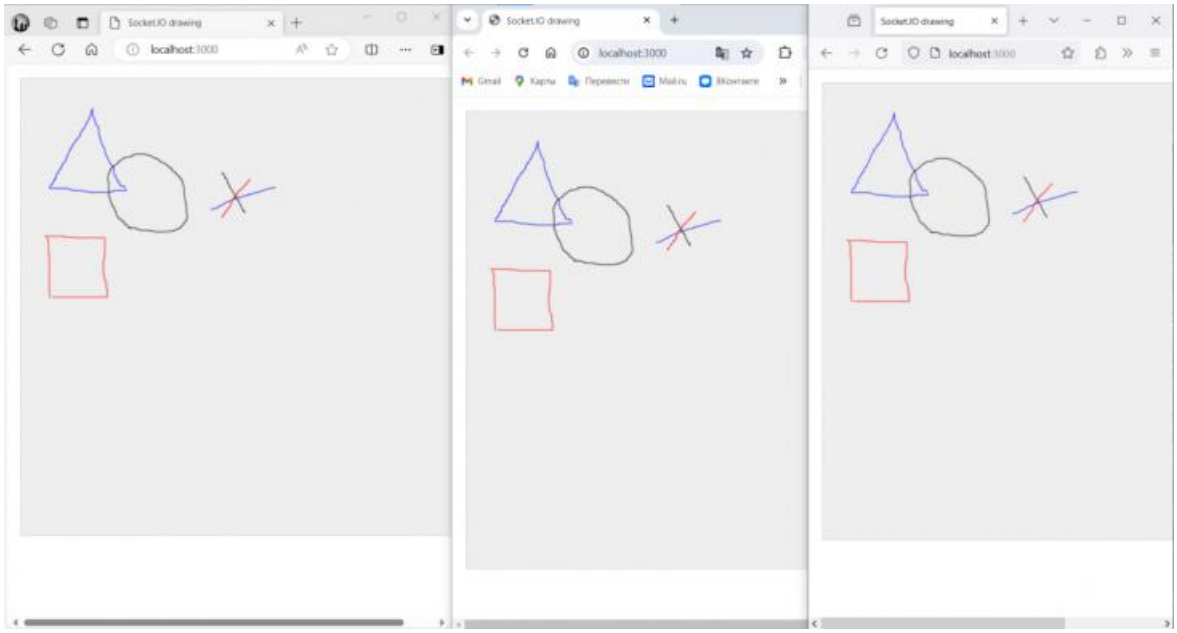


Рисунок 3. Результат работы алгоритма

Заключение

Можно сделать вывод, что с помощью библиотеки Socket.IO возможна реализация приложений совместного редактирования графических схем в режиме реального времени на основе описанного алгоритма. Предложенный в данной работе метод показал свою стабильность работы и удобство реализации. Стоит отметить, что данный алгоритм ожидает доработок в том случае, если, например, необходимо рисовать ровные графические фигуры одним движением мыши, что потребует дополнительных будущих исследований. Тем не менее, полученный алгоритм является удобной основой для более сложных программ.

Библиография

1. Князев А. А., Кондратьев А. Н., Дубровский Н. С. Эволюция и особенности протокола HTTP // Инновационный потенциал развития общества: взгляд молодых ученых: сборник научных статей 4-й Всероссийской научной конференции перспективных разработок, Курск, 01 декабря 2023 года. – Курск: ЗАО «Университетская книга», 2023. – С. 176-178.
2. Kovaliuk, D., Kovaliuk, O.O., Pinaieva, O., Kotyra, A., & Kalizhanova, A. (2019). Optimization of web-application performance. *Symposium on Photonics Applications in Astronomy, Communications, Industry, and High-Energy Physics Experiments (WILGA)*.
3. Gursesli, M.C.; Selek, M.E.; Samur, M.O.; Duradoni, M.; Park, K.; Guazzini, A.; Lanatà, A. Design of Cloud-Based Real-Time Eye-Tracking Monitoring and Storage System. Algorithms 2023, 16, 355. https://doi.org/10.3390/a16070355
4. Горчаков А. Я. Разработка клиентской архитектуры системы мгновенных сообщений по технологии WebSocket // Гагаринские чтения-2018: Сборник тезисов докладов XLIV Международной молодёжной научной конференции, Москва-Байконур-Ахтубинск, 17–20 апреля 2018 года. Том 2. – Москва-Байконур-Ахтубинск: Московский авиационный институт (национальный исследовательский университет), 2018. – С. 209.
5. Шабанов А. Э. Обзор библиотеки socket.io // Информационно-компьютерные технологии в экономике, образовании и социальной сфере. – 2022. – № 1(35). – С. 56-62.

6. Царева Е. В. Разрешение конфликтных ситуаций при синхронизации многопользовательских online-приложений // Вестник Томского государственного университета. Управление, вычислительная техника и информатика. – 2016. – № 1(34). – С. 79-91.
7. Чернышев Я. Р. Разработка и самостоятельный хостинг веб-приложения на основе фреймворка Express.js // Студенческая наука-взгляд в будущее: Материалы XVIII Всероссийской студенческой научной конференции, Красноярск, 15–17 марта 2023 года. Том Часть 5. – Красноярск: Красноярский государственный аграрный университет, 2023. – С. 291-293.
8. Karam, Sameer & Abdulrahman, Bikhtiyar. (2022). Using Socket.io Approach for Many-to-Many Bi-Directional Video Conferencing. AL-Rafidain Journal of Computer Sciences and Mathematics. 16. 81-86. 10.33899/csmj.2022.174411.
9. Macklon, Finlay & Viggiato, Markos & Romanova, Natalia & Buzon, Chris & Paas, Dale & Bezemer, Cor-Paul. (2023). A Taxonomy of Testable HTML5 Canvas Issues. IEEE Transactions on Software Engineering. PP. 1-13. 10.1109/TSE.2023.3270740.
10. Кочитов М. Е. Рисование компьютерной мышью на холсте веб-страницы браузера с помощью инструмента Canvas // Постулат. – 2019. – № 8(46). – С. 25

Результаты процедуры рецензирования статьи

В связи с политикой двойного слепого рецензирования личность рецензента не раскрывается.
Со списком рецензентов издательства можно ознакомиться здесь.

В статье авторы подробно излагают проблематику существующих методов взаимодействия в веб-приложениях, акцентируя внимание на недостатках протокола HTTP для задач совместной работы и предлагая в качестве решения использование WebSocket с библиотекой Socket.IO. Методология исследования представлена разработкой модели процесса совместного редактирования и алгоритма, реализующего эту модель в реальном времени. Особое внимание уделено обработке конфликтов при одновременном внесении изменений разными пользователями. Методология объединяет теоретический анализ и практическую разработку, демонстрируя на примере разработки веб-приложения. Актуальность исследования очевидна, учитывая растущую потребность в инструментах для совместной работы в режиме онлайн. Современные веб-технологии предлагают новые возможности для разработки подобных приложений, и поиск оптимальных решений в этой области представляет большой интерес. Научная новизна заключается в разработке конкретного алгоритма с использованием библиотеки Socket.IO для решения задачи совместного редактирования графических схем. Особенностью является подробное описание механизма обработки конфликтов, возникающих при одновременном редактировании одного документа. Статья имеет четкую структуру, включая введение, описание проблемы, разработку модели процесса, описание алгоритма и заключение. Стиль изложения научный, содержание подается логично и последовательно. Иллюстрации и схемы эффективно дополняют текст, способствуя лучшему пониманию материала. Библиография содержит актуальные источники, что свидетельствует о тщательном подходе авторов к изучению предмета. Список литературы представляет собой смесь работ по теоретическим основам и практическим разработкам в области веб-технологий. Статья будет интересна как исследователям в области компьютерных наук, так и практикующим разработчикам, заинтересованным в создании совместных редакторов и приложений реального времени. Выводы подчеркивают потенциал использования библиотеки Socket.IO для

решения задачи совместного редактирования графических схем, а также указывают на возможности дальнейших исследований в этом направлении, включая оптимизацию производительности и улучшение пользовательского интерфейса. Авторы уверенно демонстрируют, как их разработка может служить основой для более сложных и функциональных систем совместной работы. Также они отмечают необходимость дальнейших исследований для решения специфических задач, например, реализации более сложных графических фигур и интеграции с другими инструментами взаимодействия в реальном времени. Статья представляет значительный интерес для научного сообщества и практикующих специалистов в области разработки программного обеспечения для совместной работы. Материал изложен доступно, аргументированно и снабжен необходимыми примерами и иллюстрациями. Однако, авторам могут быть предложены к доработке следующие аспекты: более подробное обсуждение потенциальных ограничений предложенного решения и более глубокий анализ существующих альтернативных подходов к совместному редактированию в режиме реального времени. Это позволит усилить позицию работы в контексте современных исследований и разработок. Также рекомендую разбить рисунок 4 на несколько частей (в виде отдельных рисунков) с их детальным описанием. Исходя из вышеизложенного, рекомендую статью к публикации после выполнения незначительных доработок, указанных выше, для дальнейшего усиления аргументации и обоснованности выбранных решений.

Результаты процедуры повторного рецензирования статьи

В связи с политикой двойного слепого рецензирования личность рецензента не раскрывается.
Со списком рецензентов издательства можно ознакомиться здесь.

Предметом исследования в рецензируемой статье выступает алгоритм и программная реализация совместного редактирования графических схем в режиме реального времени с использованием библиотеки Socket.IO.
Методология исследования базируется на обобщении сведений из научных публикаций по изучаемой теме, программной реализации предлагаемого алгоритма.
Актуальность работы авторы связывают с наличием существенных недостатков в протоколе HTTP для реализации алгоритма совместного редактирования графических схем в режиме реального времени (отсутствие двунаправленной и полнодуплексной связи между компонентами системы), из-за которых процесс обмена координатами курсора мыши между сервером и клиентами при рисовании линий в браузере может вызвать задержки и оказывать дополнительную нагрузку на сервер.
Научная новизна рецензируемого исследования, по мнению рецензента, состоит в предложениях по использованию JavaScript-библиотеки Socket.IO, которая при совместном редактировании графических схем добавляет управляемость событиями, автоматическое переподключение при разрыве соединения, мультиплексирование и другие функции.
В статье структурно выделены следующие разделы: Введение, Модель процесса совместного редактирования в реальном времени, Алгоритм совместного редактирования графических схем в режиме реального времени, Заключение, Библиография.
Авторы рассматривают ситуацию, когда два пользователя работают над одной графической схемой в разных процессах, каждый пользователь вносит изменения в документ независимо от других пользователей, и все изменения синхронизируются в реальном времени. В публикации приведена серверная часть алгоритма, реализованная

с использованием минималистичного фреймворка Express. По мнению авторов, применение протокола WebSocket позволяет веб-серверу и браузеру обмениваться данными по одному и тому же соединению одновременно в обоих направлениях, сервер в любой момент времени может отправить сообщение сразу всем клиентам, ранее установившим с ним соединение. В результате исследования авторы приходят к выводу о том, что с помощью библиотеки Socket.IO возможна реализация приложений совместного редактирования графических схем в режиме реального времени на основе описанного алгоритма.

Библиографический список включает 10 источников – публикации отечественных и зарубежных ученых по теме статьи, на которые в тексте имеются адресные ссылки, подтверждающие наличие апелляции к оппонентам.

Из существенных недостатков публикации следует отметить, что рисунок 4 не читается, а судя по тексту статьи именно на нем отражены итоги исследования в виде обобщенной схемы работы предложенного алгоритма совместного редактирования графических схем в режиме реального времени.

Статья отражает результаты проведенного авторами исследования, соответствует направлению журнала «Программные системы и вычислительные методы», содержит элементы научной новизны и практической значимости, может вызвать интерес у читателей, но нуждается в доработке в соответствии с высказанным замечанием.

Программные системы и вычислительные методы
Правильная ссылка на статью:
Малахов С.В., Якупов Д.О., Воробьева Е.Г., Нехаев М.В., Мухтулов М.О., Новосельцева С.В. Развитие и применение операционных систем и оболочек в мобильных технологиях: анализ истории развития и актуальных трендов в сфере мобильных ОС и оболочек // Программные системы и вычислительные методы. 2024. № 1. DOI: 10.7256/2454-0714.2024.1.70144 EDN: VPNJNF URL: https://nbpublish.com/library_read_article.php?id=70144

Развитие и применение операционных систем и оболочек в мобильных технологиях: анализ истории развития и актуальных трендов в сфере мобильных ОС и оболочек

Малахов Сергей Валерьевич

ORCID: 0009-0001-8666-6713

кандидат технических наук

доцент, кафедра управления в технических системах, Поволжский государственный университет телекоммуникаций и информатики

443010, Россия, Самарская область, г. Самара, ул. Льва Толстого, 23

s.malakhov@psuti.ru

Якупов Денис Олегович

ORCID: 0009-0003-2371-0822

ассистент, аспирант, кафедра управления в технических системах, Поволжский государственный университет телекоммуникаций и информатики

443010, Россия, Самарская область, г. Самара, ул. Льва Толстого, 23

d.yakupov@psuti.ru

Воробьева Евгения Григорьевна

ORCID: 0009-0008-8225-7091

студент, кафедра информатики и вычислительной техники, Поволжский государственный университет телекоммуникаций и информатики

443010, Россия, Самарская область, г. Самара, ул. Л. Толстого, 23

vorobeva.g2004@gmail.com

Нехаев Максим Вадимович

студент, кафедра информатики и вычислительной техники, Поволжский государственный университет телекоммуникаций и информатики

443010, Россия, Самарская область, г. Самара, ул. Л. Толстого, 23

maks.popovich2014@yandex.ru

Мухтулов Михаил Олегович

студент, кафедра информатики и вычислительной техники, Поволжский государственный университет телекоммуникаций и информатики

443010, Россия, Самарская область, г. Самара, ул. Л. Толстого, 23

mixa.1204@inbox.ru

Новосельцева София Владимировна

студент, кафедра информатики и вычислительной техники, Поволжский государственный университет телекоммуникаций и информатики

443010, Россия, Самарская область, г. Самара, ул. Л. Толстого, 23

sunny.tea.with.lilac@gmail.com

Статья из рубрики "Языки программирования"

DOI:

10.7256/2454-0714.2024.1.70144

EDN:

VPNJNF

Дата направления статьи в редакцию:

16-03-2024

Аннотация: Объектами исследования являются мобильные операционные системы и их оболочки. В качестве предмета исследования используется функциональность операционных систем Android, iOS и HarmonyOS, их история создания и тенденции развития. Авторы подробно рассматривают такие аспекты темы как, история создания операционных систем Android, iOS, HarmonyOS и оболочек TouchWiz, HTC Sense, MIUI и других, современные тренды в области мобильных операционных систем, которые отражают влияние технологических новинок и геополитических аспектов на развитие данной сферы. Проводят развернутый анализ ОС, используя приложения для тестирования производительности (AnTutu Benchmark, 3DMark Benchmark). Целью данного исследования является изучение истории развития мобильных ОС и оболочек от истоков до современных трендов технического прогресса. Методы исследования базируются на сборе и систематизации информации, анализе и сравнении систем, а также на тестах производительности. Научная новизна данной статьи заключается в использовании ОС и оболочек в мобильных устройствах, отвечающего всем запросам и требованиям пользователя, учитывая бурное развитие цифровых технологий и всё большее внедрение их в нашу повседневную жизнь. Основными выводами проведенного исследования являются выявление самой распространенной ОС, определение современных трендов, которые включают в себя интеграцию искусственного интеллекта, мультимодальность, обеспечение безопасности и конфиденциальности, а также расширение гибкости и портативности. Стремительное развитие технологий и вселенной мобильных приложений делает мобильные ОС и оболочки ключевыми компонентами успешного пользовательского опыта в мире мобильных технологий. Осознание истории и актуальных тенденций в сфере мобильных операционных систем и оболочек позволит более точно предугадать технологические изменения и потенциальные влияния в будущем.

Ключевые слова:

оболочка, устройство, система, открытый исходный код, операционная, искусственный интеллект, мировой рынок, мобильная ОС, операционная система, программное обеспечение

Введение

В последние десятилетия операционные системы и оболочки стали неотъемлемой частью мобильных технологий, играя ключевую роль в функциональности и пользовательском опыте мобильных устройств. Смартфоны и планшеты позволяют нам быть всегда на связи, получать информацию, развлекаться и работать в любой точке мира. Одним из ключевых факторов успешной работы мобильных устройств являются операционные системы (ОС) и оболочки.

Операционная система (ОС) – это программное обеспечение, которое управляет и координирует работу мобильного устройства. ОС обеспечивает связь между аппаратными и программными компонентами устройства, обеспечивает взаимодействие с пользователем и управляет выполнением приложений.

В свою очередь, оболочка – это пользовательский интерфейс ОС, который предоставляет доступ к функциональности устройства и приложений. Оболочка включает в себя элементы управления и интерактивные компоненты, такие как меню, панели инструментов, значки и жесты, упрощающие взаимодействие пользователя с устройством.

В данной статье рассматривается история развития мобильных ОС и оболочек, а также выявляются актуальные тренды в этой сфере.

1. История развития ОС и оболочек

История развития мобильных операционных систем восходит к появлению первых коммерческих смартфонов в начале 2000-х годов.

Первая мобильная ОС: Symbian. Разработанная компанией Psion, Symbian стала одной из первых операционных систем, предназначенных специально для мобильных устройств. Она была наиболее популярной в то время. В 2007 году Apple представила свою мобильную ОС iOS для iPhone. Ее основателем был Стив Джобс. Операционная система была инновационной и привлекла внимание миллионов пользователей по всему миру и сильно изменила игру в мобильной индустрии. История Android началась с 2003 года, ее основателями были Рич Майнер, Ник Сирс, Крис Уайт и Энди Рубин. Компания Google, после приобретения ОС Android, начинает активно развивать эту мобильную платформу, которая впоследствии становиться самой популярной ОС в мире, благодаря своей открытости, широкой поддержке и богатым возможностям настройки. [1] Microsoft также внесла свой вклад в мир мобильных ОС с выпуском Windows Phone. Однако эта платформа не смогла конкурировать с Android и iOS и в итоге была приостановлена. В 2019 году компания Huawei представила свою собственную ОС – HarmonyOS, но доступна она только на устройствах Huawei и Honor. Также начиная с 2016 года находится в разработке Российская ОС «Аврора».

С того времени операционные системы, такие как iOS, Android и другие, претерпели значительные изменения и улучшения, приведшие к более продвинутым возможностям и интерфейсам для мобильных устройств.

Важной частью развития мобильных операционных систем стало применение оболочек. В начале 2000-х оболочки представляли собой простые интерфейсы, предоставляющие доступ к основным функциям, таким как звонки, SMS и некоторые игры. С появлением iOS и Android оболочки стали более интерактивными и гибкими. Они начали включать магазины приложений, настраиваемые виджеты и широкие возможности настройки. Примером таких оболочек являются: TouchWiz, HTC Sense и MIUI, предоставляющие уникальные функции и дизайн, дополняющие базовую функциональность операционных систем. С развитием и появлением новых операционных систем, потребовались и усовершенствованные оболочки, такие как EMUI, One UI и другие.

2. Анализ популярных ОС

Далее будет проведен анализ самых популярных операционных систем. Для сравнения используем: iOS, Android и HarmonyOS.

Android является самой распространенной ОС в мире. Она предназначена для устройств различных производителей независимо от характеристик и цен. ОС имеет открытый исходный код, что позволяет разработчикам создавать новые приложения и функции, доступные для скачивания на Google Play Store, а также дает возможность устанавливать приложения из сторонних источников. Еще один плюс системы – это более гибкие настройки интерфейса и функций устройства. Android совместим с большим разнообразием аксессуаров.

iOS является второй по популярности системой, которая предназначена только для устройств компании Apple. Данная ОС имеет закрытый исходный код, что обеспечивает высокую безопасность и стабильную работу устройств. Apple выпускает обновления программного обеспечения чаще, чем другие производители операционных систем.[2]

HarmonyOS одна из новых операционных систем, для устройств компании Huawei. Она имеет открытый исходный код как у системы Android, но у нее улучшенная безопасность как у iOS, за счет использования собственных механизмов защиты и шифрования данных.

По данным сайта «Исследование рынка технологий Counterpoint», распространение мобильных ОС на мировом рынке в 2023 году (Рис.1) в процентном соотношении составил: Android – 81% , iOS – 16% и HarmonyOS –3%.[4]

Рисунок 1. Распространение мобильных ОС на мировом рынке в 2023 году

Рассмотрев преимущества и недостатки операционных систем, можно сделать вывод, что Android имеет открытый исходный код для разработки приложений и применяется на многих устройствах. А iOS и HarmonyOS обладают высокой степенью безопасности и обеспечивают стабильную работу устройств, но только в рамках одного бренда.

3. Исследование производительности ОС

Сравнение производительности мобильных операционных систем – это всегда актуальная тема для пользователей смартфонов. Несмотря на то, что многие предпочтения зависят от личных предпочтений и потребностей, важно понимать различия между разными ОС. Рассмотрим, как Android, iOS и HarmonyOS выступают в тестах производительности.

В тестах использовались модели смартфонов: Samsung Galaxy S20 (Android), iPhone 12 Pro (iOS), Huawei P40 Pro (HarmonyOS). Для тестирования применялись следующие приложения:

AnTuTu Benchmark для Android - приложение для тестирования производительности смартфонов и планшетов. Включает тесты производительности графики, вычислений искусственного интеллекта, тест браузера, а также функции проверки экрана, памяти и батареи.[5]

3DMark Benchmark – популярный тест производительности, который позволяет протестировать производительность графического процессора и центрального процессора устройства.[6]

1. Тест производительности в Antutu Benchmark (Рис.2):

Samsung Galaxy S20 (Android):

· Общий балл (Total Score): Средний результат от 400 000 до 500 000.

· Балл процессора (CPU Score): Средний результат от 100 000 до 150 000. Samsung Galaxy S20 оснащается мощным процессором Snapdragon 865, обеспечивающим высокую производительность в многозадачных приложениях и играх.

· Балл графики (GPU Score): Средний результат от 150 000 до 180 000. Устройство обладает высококлассной графической подсистемой, которая способна обеспечить плавный игровой опыт с высоким разрешением и частотой обновления экрана.

· Балл памяти (Memory Score): Средний результат от 60 000 до 80 000. Samsung Galaxy S20 имеет достаточно оперативной памяти для эффективной работы с множеством приложений и процессов одновременно.

· Балл UX (User Experience Score): Средний результат от 70 000 до 100 000. Этот показатель отражает качество пользовательского опыта, включая скорость работы интерфейса, быстродействие приложений и другие факторы.

iPhone 12 Pro (iOS):

· Общий балл (Total Score): Средний результат от 600 000 до 700 000. iPhone 12 Pro оснащен мощным процессором Apple A14 Bionic и высокопроизводительной графикой, что обеспечивает впечатляющую общую производительность.

· Балл процессора (CPU Score): Средний результат от 180 000 до 220 000. Процессор A14 Bionic имеет шесть ядер, что обеспечивает быструю обработку данных и

многозадачность.

· Балл графики (GPU Score): Средний результат от 200 000 до 250 000. Встроенный графический процессор обеспечивает высокую производительность для графических приложений и игр.

· Балл памяти (Memory Score): Средний результат от 80 000 до 100 000. iPhone 12 Pro оснащен 6 ГБ оперативной памяти, что обеспечивает быструю загрузку и многозадачность.

· Балл UX (User Experience Score): Средний результат от 130 000 до 150 000.

Huawei P40 Pro (HarmonyOS):

· Общий балл (Total Score): Средний результат от 450 000 до 550 000.

· Балл процессора (CPU Score): Средний результат от 120 000 до 150 000. Huawei P40 Pro обычно оснащен мощным процессором Kirin 990 с технологией 7 нм, что обеспечивает высокую производительность.

· Балл графики (GPU Score): Средний результат от 180 000 до 200 000. Графический процессор Mali-G76 MP16 в сочетании с Kirin 990 обеспечивает отличную графическую производительность..

· Балл памяти (Memory Score): Средний результат от 80 000 до 100 000. Huawei P40 Pro обычно оснащен 8 ГБ оперативной памяти, что обеспечивает хорошую производительность при многозадачности.

· Балл UX (User Experience Score): Средний результат от 70 000 до 100 000.

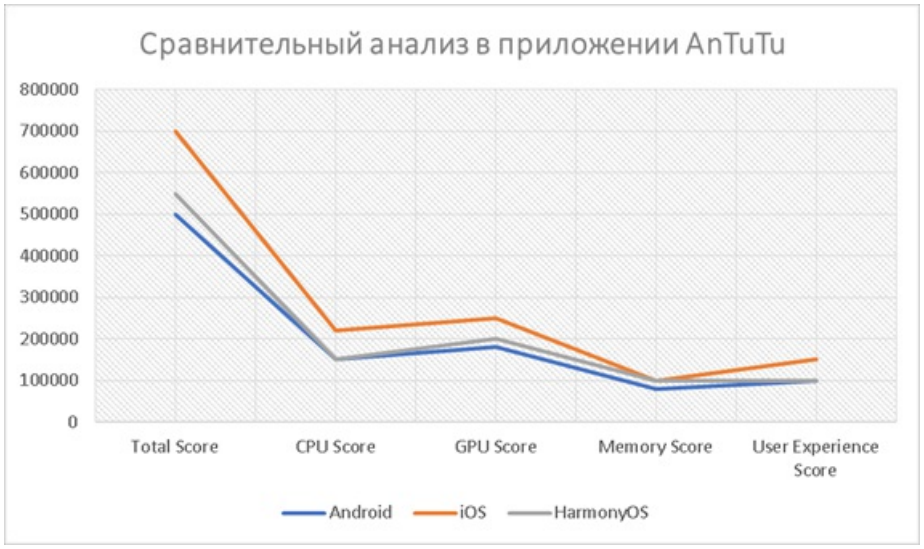


Рисунок 2. Тест производительности в Antutu Benchmark

2. Тест производительности графики в 3DMark Benchmark (Рис.3):

Samsung Galaxy S20 (Android):

· Wild Life (Graphics Test): Средний результат от 6000 до 8000. Wild Life тестирует графическую производительность устройства в играх с высоким разрешением и сложными эффектами.

· Sling Shot (Graphics Test): Средний результат от 7000 до 9000. Sling Shot тестирует

графическую производительность в более современных мобильных играх с использованием различных графических API.

· Ice Storm (Graphics Test): Средний результат от 50 000 до 70 000. Ice Storm предоставляет результаты теста графической производительности в менее требовательных играх и приложениях.

iPhone 12 Pro (iOS):

· Ice Storm Unlimited (Graphics Test): Средний результат от 120 000 до 140 000.

· Sling Shot Extreme (Graphics Test): Средний результат от 6000 до 7000.

· Wild Life (Graphics Test): Средний результат от 6000 до 7000.

Huawei P40 Pro (HarmonyOS):

· Ice Storm Unlimited (Graphics Test): Средний результат от 100 000 до 120 000.

· Sling Shot Extreme (Graphics Test): Средний результат от 4000 до 5000.

· Wild Life (Graphics Test): Средний результат от 5000 до 6500.

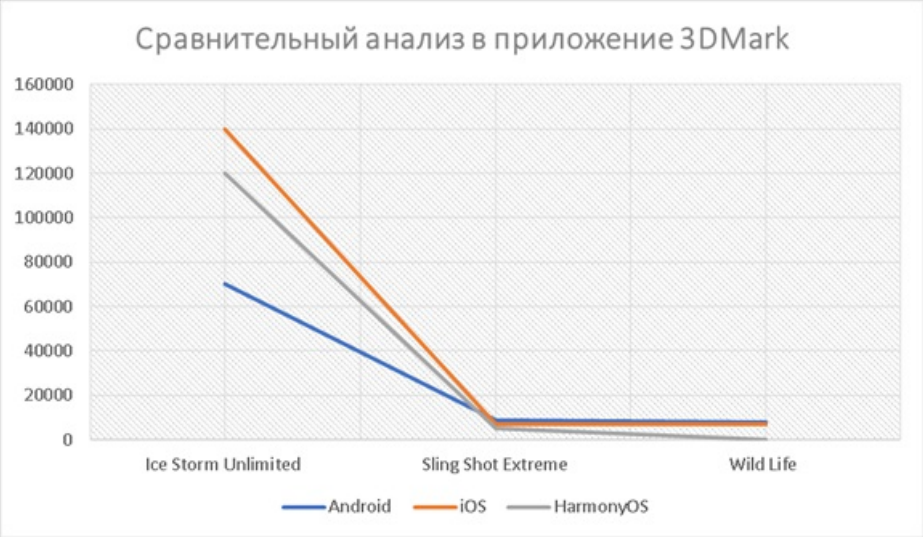


Рисунок 3. Тест производительности графики в 3DMark Benchmark

3. Тест скорости работы приложений (Рис.4):

· Android: Среднее время отклика 0,6 секунды

· iOS: Среднее время отклика 0,8 секунды

· Harmony OS: Среднее время отклика 0,7 секунд

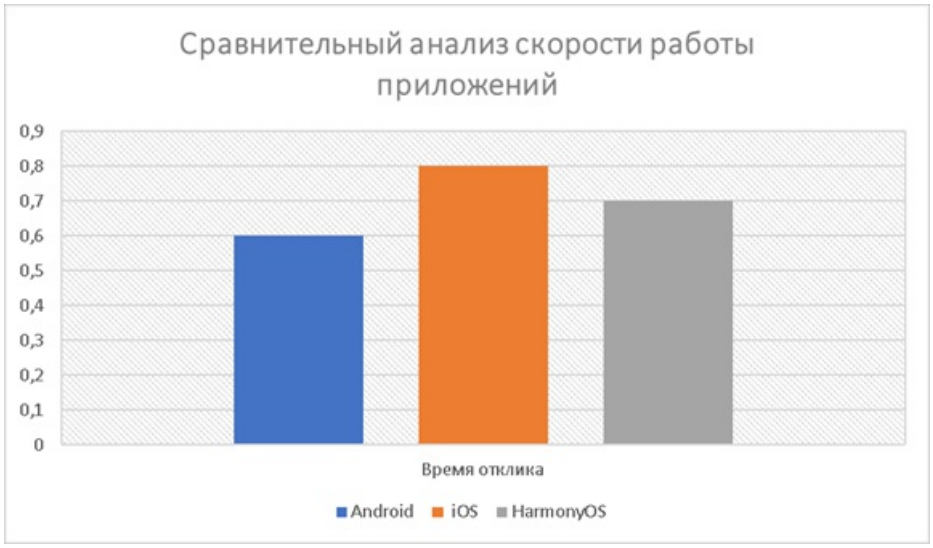


Рисунок 4. Тест скорости работы приложений

Исходя из проведенных тестов, можно сделать вывод, что iOS и HarmonyOS обладают схожей производительностью, превосходя Android в некоторых аспектах. iOS показывает лучшие результаты, однако, стоит отметить, что HarmonyOS еще относительно новая операционная система и ее производительность может улучшаться с развитием и оптимизацией.

4. Тенденции развития ОС и оболочек

Современные тренды в области мобильных операционных систем отражают влияние технологических новинок и геополитических аспектов на развитие данной сферы. Самое популярное и многообещающее направление – интеграция искусственного интеллекта (ИИ). Многие современные мобильные ОС и оболочки внедряют ИИ-технологии для улучшения пользовательского опыта. Это позволяет создавать персонализированные рекомендации, улучшать распознавание голоса и изображений, а также повышать безопасность устройств.

Еще одним актуальным трендом является мультимодальность. Создание оболочек и ОС, которые обеспечивают возможность взаимодействия с устройствами различными способами: голосом, жестами, касанием и т.д.

Производители постоянно ищут способы улучшить пользовательский интерфейс, делая его более интуитивно понятным и удобным для пользователя. Большое внимание уделяется графическому дизайну и анимации.

В современном мире безопасность данных является приоритетной задачей. Поэтому с увеличением угроз кибербезопасности, разработчики уделяют большое внимание защите данных пользователей, внедряя новые методы шифрования, биометрическую аутентификацию и другие меры безопасности.

Увеличение производительности и оптимизации. С появлением более мощных мобильных устройств, разработчики ОС и оболочек стремятся к оптимизации кода и повышению производительности, чтобы обеспечить более быструю и плавную работу устройств.

Интеграция с другими устройствами и сервисами. Мобильные ОС и оболочки все больше интегрируются с другими устройствами (умные часы, домашние устройства и т. д.) и онлайн-сервисами для создания более удобной и совместимой экосистемы.

Расширение возможностей гибкости и портативности. С появлением смартфонов с гибкими экранами и смарт-часов, мобильные ОС и оболочки также разрабатываются с учетом возможности адаптации к различным формам и факторам. Это позволяет пользователям настроить устройства в соответствии с их потребностями и предпочтениями.[3]

В целом, тенденции развития в сфере операционных систем и оболочек в мобильных технологиях указывают на улучшение производительности, безопасности, удобства и персонализации пользовательского опыта, что позволит создавать более интеллектуальные, адаптивные и эффективные устройства.

Заключение

Таким образом, мобильные ОС и оболочки играют решающую роль в функциональности и удобстве использования мобильных устройств. Их развитие и применение тесно связаны с требованиями и потребностями пользователей. Современные тренды включают в себя интеграцию искусственного интеллекта, мультимодальность, обеспечение безопасности и конфиденциальности, а также расширение гибкости и портативности. Стремительное развитие технологий и вселенной мобильных приложений делает мобильные ОС и оболочки ключевыми компонентами успешного пользовательского опыта в мире мобильных технологий. Осознание истории и актуальных тенденций в сфере мобильных операционных систем и оболочек позволит более точно предугадать технологические изменения и потенциальные влияния в будущем. Будущее мобильных технологий обещает еще больше инноваций и удобства для пользователей.

Библиография

1. Ахметов А. К. Операционная система Android: история создания и развития. Разработка приложений для платформы Android // Скиф. Вопросы студенческой науки. 2017. №9. С. 2-3.
2. Погорелов Д. В., Колоколов Е. А., Ермолаев В. В. Сравнение мобильных операционных систем Android и ios // Вестник науки. 2022. №12 (57) Т. 5. С. 120-124.
3. Староверова Н. А., Морозов Д., Калаев И., Кадырова Г. Современные тенденции и перспективы развития операционных систем // Вестник Казанского технологического университета. 2015. Т. 18. №21. С. 134-136.
4. Сравниваем лучшие операционные системы для смартфонов в 2023 году [Электронный ресурс]. URL: https://blog.eldorado.ru/publications/battl-os-sravnivaem-luchshie-operatsionnye-sistemy-dlya-smartfonov-v-2023-godu-39758 Дата обращения: 10.03.2024.
5. AnTuTu Benchmark [Электронный ресурс]. URL: https://www.antutu.com/en/download.htm Дата обращения: 10.03.2024.
6. 3DMark [Электронный ресурс]. URL: https://benchmarks.ul.com/3dmark-android Дата обращения: 10.03.2024

Результаты процедуры рецензирования статьи

В связи с политикой двойного слепого рецензирования личность рецензента не раскрывается.
Со списком рецензентов издательства можно ознакомиться здесь.

Статья посвящена анализу истории развития и актуальных трендов в сфере

операционных систем (ОС) и пользовательских оболочек для мобильных технологий. Авторы рассматривают ключевые этапы эволюции мобильных ОС, от первых коммерческих смартфонов до современных решений, а также изучают новейшие направления в развитии мобильных интерфейсов и функциональности.

В статье применяется аналитический подход к изучению исторического развития и текущего состояния мобильных ОС и оболочек. Анализируются различные источники, включая научные публикации, отчеты рынка и технические спецификации, для выявления основных трендов и перспективных направлений развития.

Тема актуальна из-за постоянно растущего спроса на мобильные устройства и высокой конкуренции между разработчиками ОС и оболочек. Разработка удобных, функциональных и безопасных мобильных систем является ключевым фактором в обеспечении успешности смартфонов и планшетов на рынке.

Новизна статьи заключается в комплексном рассмотрении эволюции мобильных ОС и оболочек, начиная от их зарождения до современного состояния. Особое внимание уделено анализу последних трендов, таких как интеграция искусственного интеллекта, мультимодальность, повышение безопасности и оптимизация интерфейсов.

Статья написана грамотным научным стилем, содержит все необходимые разделы: введение, обзор истории развития ОС, сравнение популярных ОС, исследование производительности, анализ текущих трендов и заключение. Такой подход обеспечивает удобство чтения и понимания материала.

Однако к статье есть замечания:

1) Оценка различных операционных систем на разных аппаратных устройствах может быть как правильной, так и спорной практикой Подход к оценке различных ОС на разных аппаратных устройствах требует тщательного планирования и ясного понимания целей исследования. Важно обеспечить, чтобы сравнение было справедливым и что различия в производительности можно было корректно приписать особенностям ОС, а не аппаратным различиям. Также крайне важно прозрачно сообщать об этих ограничениях при представлении результатов.

2) В текущей версии статьи представлены важные, но возможно ограниченные источники. Для углубления анализа и обогащения контекста исследования, а также для подтверждения сделанных выводов и обеспечения большей научной обоснованности, авторам рекомендуется включить релевантные виды источников:

- включение работ, опубликованных в последние годы, поможет подчеркнуть актуальность темы и показать последние тенденции в развитии мобильных ОС и оболочек;
- исследования, сравнивающие различные ОС на однотипных аппаратных устройствах, могут дать более глубокое понимание преимуществ и недостатков каждой системы;
- включение ссылок на документацию от Apple (iOS), Google (Android) и других разработчиков ОС может предоставить авторитетные сведения о функциональных возможностях и архитектуре систем;
- ссылки на стандарты и протоколы безопасности, используемые в ОС, помогут подчеркнуть аспекты безопасности, обсуждаемые в статье;
- данные от аналитических компаний, таких как Gartner, IDC, Counterpoint, могут предоставить объективную статистику и тенденции рынка, подтверждающие утверждения авторов о популярности и распространенности различных ОС.
- включение исследований, посвященных пользовательскому опыту и удовлетворенности различными ОС, может обогатить анализ предпочтений пользователей и влияния интерфейсов на удобство использования.
- материалы с профильных конференций и семинаров по мобильным технологиям и информационной безопасности могут представлять интерес для подтверждения

сделанных выводов и обсуждения будущих направлений развития.

Расширение библиографического списка таким образом не только усилит научную обоснованность статьи, но и повысит ее ценность для читателей, заинтересованных в глубоком понимании темы.

Исправление этих недостатков поможет сделать статью более ценной для научного сообщества и читателей, заинтересованных в развитии мобильных технологий и операционных систем.

Программные системы и вычислительные методы
Правильная ссылка на статью:
Дагаев Д.В. Инструментальный подход к программированию в системе МультиОберон // Программные системы и вычислительные методы. 2024. № 1. DOI: 10.7256/2454-0714.2024.1.69437 EDN: WVZVVU URL: https://nbpublish.com/library_read_article.php?id=69437

Инструментальный подход к программированию в системе МультиОберон

Дагаев Дмитрий Викторович

ORCID: 0000-0003-0343-3912

Генеральный директор, ООО "СКАДИ"

115230, Россия, г. Москва, Зеленый проспект, 5/12, строение 3, пом. 1

dvdagaev@oberon.org

Статья из рубрики "Языки программирования"

DOI:

10.7256/2454-0714.2024.1.69437

EDN:

WVZVVU

Дата направления статьи в редакцию:

25-12-2023

Аннотация: Объектно-ориентированные подходы к программированию, имеют свою область применимости. Для ряда задач традиционно отдают предпочтение классическим методам структурного программирования. Эти предпочтения нередки для детерминированного мира и в системах, ориентированных на машинное представление. Исторически классические методы развивались от архитектуры фон Неймана представления машины. При решении проблем детерминированного мира выявляются преимущества подходов, противоположных объектно-ориентированному мышлению. Например, язык и системы на базе модульного языка программирования Оберон в классической реализации демонстрируют минималистский путь для достижения надежности, существенно отличающийся от большинства программных систем, стремящихся к максимизации числа поддерживаемых функций. Технология программирования, управляемого данными также отходит от традиционной объектной модели, требуя сепарации кода от данных. Предлагаемый автором статьи инструментальный подход объединяет Оберон-технологии с программированием, управляемым данными, при этом оставляя присущие для ООП механизмы

взаимодействия по интерфейсам. Вместо объекта предложен ассоциированный с объектом инструмент, не сохраняющий в себе данные. Предложенный в данной статье инструментальный подход отличается как от объектного представления, так и от классического структурного. Он позволяет сохранять преимущества обоих подходов. При этом инструментальный подход работает в инфраструктуре программирования управляемого данными. От объектно-ориентированного подхода берется полиморфизм и возможность работы по интерфейсам. От классического структурного программирования берется определение структур данных и взаимодействие с ними. От программирования, управляемого данными используется сепарация кода от данных и жизненный цикл последних в персистентном виде. Новизной является то, что инструментальный подход предлагает отличную от ООП ветвь развития для классического языка программирования Оберон и классического подхода. Реализованные в системе МультиОберон, инструментальный подход позволяет решать ряд важных задач, в частности, задачи автоматизации в критически важных системах.

Ключевые слова:

инструментальный подход, Оберон, ограничение, компилятор, модульность, программирование управляемое данными, МультиОберон, СКАДА, Информатика-21, Метаданные

Введение

Объектно-ориентированные подходы к программированию, равно как и другие методы, имеют свою область применимости. Для одних задач осмысливание видов «все является объектом», «думай как объект» является естественным способом построения архитектуры. Для других задач объектные построения представляют собой лишний уровень абстракций, не имеющий явного отношения к решаемой проблеме, но имеющий явно выраженные стоимостные показатели в виде избыточности компьютерных ресурсов.

Следует отметить, что даже сторонники ООП признают локальность предлагаемых ими решений, например в [1] мир разделяется на области, где объектная парадигма и парадигма классического структурного программирования разделяются по осям «знания-социальный мир» и «реализация-детерминированный мир». С этим утверждением согласны разработчики областей детерминированного мира, отдающие предпочтение классическим подходам на основе алгоритмов и структур данных [2]. При этом важным направлением являются достижения объектного мира в части абстрактного доступа через интерфейсы и организации взаимодействия.

Язык программирования и система Оберон [3] исторически были основаны на минималистском классическом подходе, при наличии полиморфизма в расширении типов данных. В дальнейшем Оберон-2 развивался в виде добавления методов, а семейство Оберон-07 развивалось в сторону исключения объектной, динамической функциональности для последующего применения в контроллерах и иных встроенных системах.

Автором предложен и используется подход ограничительной семантики, который вместо нескольких языков и компиляторов использует один язык и фронтенд компилятора с системой семантических ограничений. МультиОберон представляет собой программную реализацию компилятора языка Оберон с системой семантических ограничений [4].

Решения на МультиОбероне задач из детерминированного мира потребовали ограничить имеющуюся в языке семантику работы с объектами и использовать классические структуры данных. При этом возникают вопросы интерфейсов и абстракций для доступа к данным.

В данной статье предложен инструментальный подход, при котором вместо интерфейса объекта, содержащего данные неизвестной структуры, используется интерфейс инструмента, не содержащего в себе данных и связанного с экземпляром данных. Инструмент, в отличии от объекта, не аллокируется в динамической памяти, не требует сериализации/десериализации, не может потерять данных при аварийном завершении из-за исключительной ситуации.

Понятие инструмента прозрачно, если взять аналогию с созданием скульптуры через обработку на разных фазах разными инструментами от черновой до финишной отделки.

Инструментальный подход хорошо согласуется с дата-центричным программированием, когда программы отделены от данных и связи между каждым экземпляром структуры данных и программой могут мгновенно и однозначно устанавливаться.

Инструментальный подход особенно выгодно использовать в системах 24x7 детерминированного мира, где время жизни данных может существенно превышать время жизни программ, если в системе организованы персистентные данные в разделяемой памяти.

Объектное мышление против структурного программирования

Про преимущества ООП было сказано достаточно, гораздо реже говорилось об областях эффективного применения. И эти области эффективного применения не охватывают 100% возникающих задач. Кроме этого, при использовании объектных подходов меняется технологическая культура производства ПО, основанная на совершенно других абстракциях, нежели классические подходы. Абстрагирование от структур данных и сосредоточение на интерфейсах как неких правилах несет несомненно преимущества в части более систематического представления каждой предметной области. При этом объект становится некоторой универсальной сущностью, но при этом теряет свои уникальные особенности в части структур данных. Следовательно, решение об использовании или неиспользовании ООП имеет смысл применять на самых ранних стадиях создания программных систем.

Чем руководствоваться и какие при этом есть варианты? Д.Херш в [1] рассматривает оси «знания-социальный мир» и «реализация-детерминированный мир» (рисунок 1).

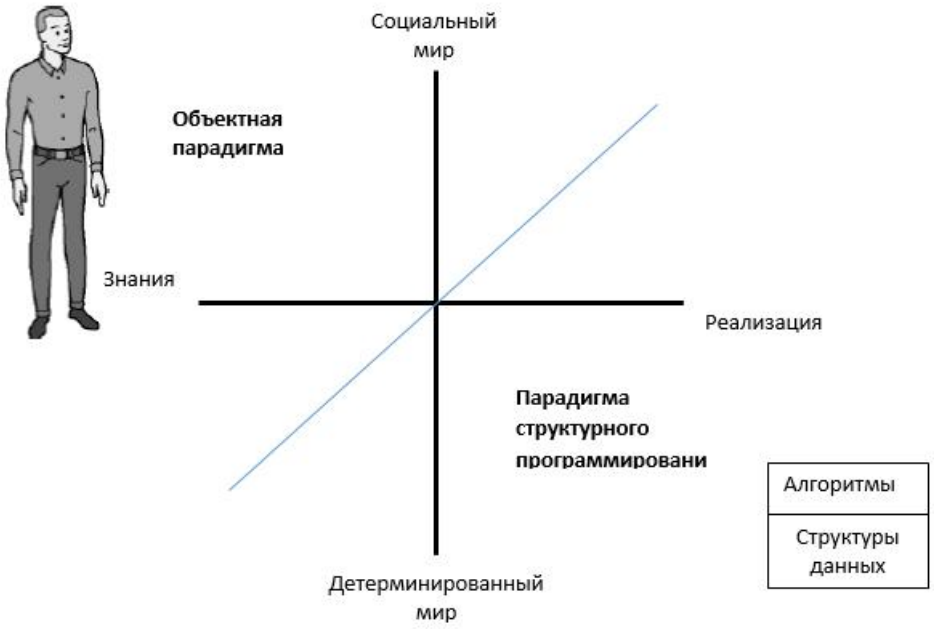


Рис. 1. Оси знания-социальный мир и реализация-детерминированный мир

Fig.1 Comprehension-Sociocultural world and Implementation-Computer Science

Разделение областей использования ООП для знаний и социального мира и классического структурного программирования для систем детерминированного мира или тех, в которых превалирует реализация, представляется логичным, особенно, учитывая, что это сформулировал теоретик ООП.

В данной статье мы будем рассматривать подходы к реализации систем детерминированного мира.

Программирование, управляемое данными

Для детерминированных программ характерна первичность данных и состояния, в таких задачах часто применяются технологии программирования, управляемого данными (data-oriented programming DOP), основные принципы которой изложены в [5].

Программы, управляемые данными, предназначены для построения слабосвязанных (loose-coupling) систем, использующих механизмы восстановления. Принципы программирования, управляемого данными диктуют следующие решения.

1. Предоставление данных и метаданных. Прямо противоположно инкапсуляции в ООП – данные о состоянии ПО предоставляются в рамках прав доступа программных компонентов. Метаданные несут информацию об имени, типе и далее о составе входящих в данный тип вложенных типов. Такое представление должно обеспечивать достаточную информацию для чтения и выполнения представленных данных формальной машиной абстрактного исполнителя.

2 . Скрытие подробностей кода. DOP скрывает программный код и основывается на сообщениях между компонентами. Интерфейс являет некритическим свойством системы. Все, что мы знаем, что имеются некие слабо связанные системы в распределенной конфигурации. В отличие от ООП зависимости между объектами и, как следствие, взаимное влияние отсутствует.

3. Сепарация кода от данных является также принципом DOP.

4 . Обработка данных должна осуществляться на основе метаданных реализующей платформы.

Важной чертой программирования, управляемого данными, является персистентность данных. Время жизни данных становится больше времени жизни программного кода. Даже при сбоях и восстановлении программного обеспечения не требуется восстановление данных.

Рисунок 2 иллюстрирует сбой и восстановление данных в ООП-системе и в DOP-системах вверху и внизу соответственно. Сбой приводит к исключению и завершению работы некоторого компонента программного обеспечения. В случае ООП при сбое возникает необходимость сериализации данных во временное хранение, что корректно реализуется достаточно редко. При восстановлении должна осуществляться десериализация данных. После восстановления должны восстанавливаться связи у объектов, адресующих данный. Все указанные механизмы являются достаточно проблемными в части реализации, особенно по событию в виде сбоя программы.

В случае DOP осуществляется аварийное завершение и рестарт программного модуля. Других действий не требуется, при условии сепарации кода от данных. Восстановление связей у объектов, адресующих данный, не требуется, т.к. все связи между объектами обеспечены метаданными реализующей платформы и находятся в персистентной памяти.

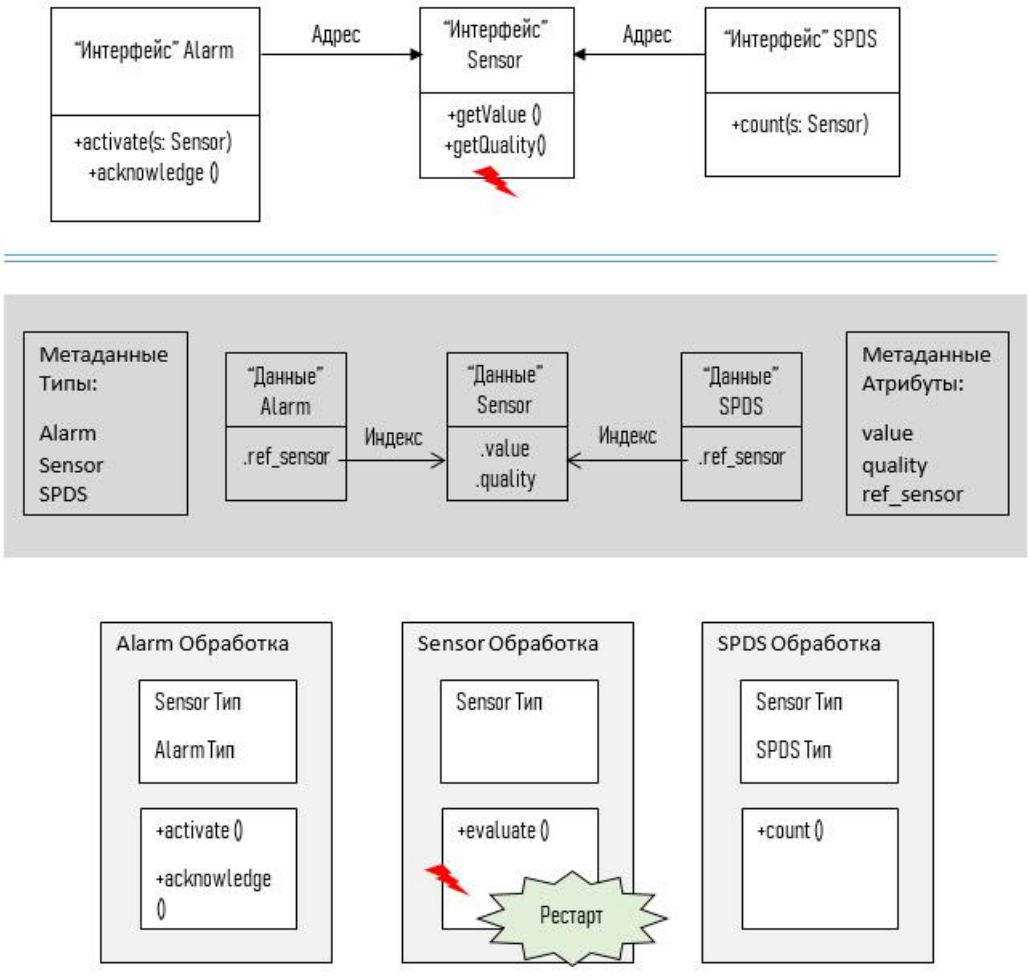


Рис. 2. Сохранение и восстановление данных ООП и DOP

Fig.2 OOP and DOP дата saving and retrieving

Сепарация кода от данных также означает сепарацию хранения, когда данные хранятся в персистентной памяти, а коды в памяти, в которую загружается программный модуль.

Семантические ограничения МультиОберона

Реализация загрузки и перезагрузки программного обеспечения может быть обеспечена либо на уровне рестарта процесса, либо средствами языка, поддерживающего модульное программирование. Оберон[3] является языком модульного программирования. Каждый программный модуль представляет собой единицу хранения, компиляции, загрузки и исполнения.

Построение систем с повышенными требованиями к функциональной безопасности приводят к запретам выделения и использования динамической памяти. В системе МультиОберон[4] указанные запреты означают, что в модуле непосредственно функция NEW быть вызвана не может. Следовательно аллокирование динамической памяти декларируется как отсутствующее. Разумеется, такие определения должны охватывать все рассматриваемые модули. Помимо рассмотрения ограничений импортируемого модуля, можно запретить использование системных преобразований, использующих данные из внешних функций.

RESTRICT –NEW, -SYSTEM;

При наличии ключевых слов NEW и SYSTEM проверка не будет пройдена и завершится синтаксической ошибкой.

Инструментальный подход вместо объектного

Отказ от динамического аллокирования приводит к изменениям в механизме реализации методов программ. Первый способ, очевидно, должен использовать подходы классического структурного программирования. Второй способ может использовать статические объекты без передачи и присваивания как указатели. Оба указанные способа теряют присущую ООП способность выстраивать системы и паттерны по интерфейсам[6], а реализации подставлять, как соответствие определенного интерфейса.

Автором предлагается инструментальный подход, заключающийся в создании вместо объекта инструмента, не содержащего в себе данные. Указатель на инструмент, подобно указателю на объект, может использоваться при подстановке и присваиваниях как данному, так и всем базовым типам. Указатель на инструмент адресует тип данных, находящейся в памяти модуля. В таблице 1 приведены отличия инструмента от объекта.

	Указатель на объект	Указатель на инструмент
Доступ по интерфейсу	Да	Да
Наличие данных	Есть	Нет
Ассоциирование с данными	Нет	Возможны
Расположение в памяти	Динамическая	Статическая
Число экземпляров	N – число объектов	Один

Таб. 1. Отличия инструмента от объекта

Table 1. Tool and object difference

Получение инструмента системной функцией TOOL

Системная функция TOOL с одним аргументом предназначена для запроса инструмента. Например, функция TOOL возвращает stdlog - указатель на инструмент типа StdLog.Hook. Инструмент представляет собой запись, не содержащую полей данных. Указатель на инструмент используется как интерфейс консоли вывода информации.

TOOL(stdlog);

stdlog.String("Tool for an object"); stdlog.Ln;

Тип инструмента должен быть определен в секции переменных.

VAR stdlog: StdLog.Hook;

Повторный и все последующие вызовы системной функции TOOL приведут в возврату того же адреса.

TOOL(stdlog2); ASSERT(stdlog=stdlog2);

Поскольку тип StdLog.Hook содержит методы и не содержит данных, то нет необходимости в выделении дополнительной памяти под каждый экземпляр инструмента. В этом заключается основное отличие системной функции TOOL от системной функции NEW.

Рис. 3. Получение адреса операторами TOOL и NEW

Fig.3 Retrieving and address by TOOL and NEW operators

Определения типов структур в языке Оберон содержатся в метаданных, имеющихся для каждого программного модуля (рисунок 3). В частности, в метаданных определен массив наследуемых типов, начиная с самого базового и заканчивая данным типом. Указатель на инструмент содержит адрес следующей за тэгом типа ячейки. Содержимое адресуемой ячейки не имеет значения, т.к. инструмент не содержит полей данных. Адресуемая ячейка располагается в области метаданных модуля. Каждый вызов TOOL возвращает адрес одной и той же ячейки.

В свою очередь, объект с данными в динамической памяти также должен содержать тэг типа и возвращать адрес следующей за тэгом типа ячейки. Па рисунке объект содержит динамические данные «Поле 1» и «Поле 2».

Использование стандартного механизма адресации с тэгом типа позволяет работать с

указателями на инструменты как с указателями на динамически аллокируемые объекты.

Однако есть важное различие между этими указателями. Указатели на динамические объекты обрабатываются сборщиком мусора после завершения использования. К указателям на инструменты сборщик мусора неприменим. Поэтому в определении указателя на инструменты должно присутствовать ключевое слово tool. Например, для нашего примера с консолью вывода информации.

Hook = POINTER TO RECORD [tool] (Log.Hook) END;

Использование системной функции TOOL допустимо только для указателей на инструменты.

Расширение базового типа инструмента

Тип данных StdLog.Hook включает в себя указатель на инструмент и используемые методы. В нашем примере он является реализацией и расширением базового типа Log.Hook. Базовый тип является абстрактным. Его определения задают интерфейсы, необходимые к реализации.

MODULE Log;

RESTRICT +TOOL;

Hook = POINTER TO ABSTRACT RECORD [tool] END;

PROCEDURE (log: Hook) String* (IN str: ARRAY OF CHAR), NEW, ABSTRACT;

PROCEDURE (log: Hook) Ln*, NEW, ABSTRACT;

Инсталляция инструмента типом StdLog обязывает реализовать в кодах все абстрактные функции, чтобы имелась возможность подстановки вместо абстрактного интерфейса log реализацию log := stdlog. Тогда каждый вызов log.String() будет приводить к вызову stdlog.String(), а архитектуру программного обеспечения можно будет выстраивать по паттернам и интерфейсам.

При реализации другой версии, например, в модуле MyStdLog, указателю будет присвоено другое значение log := mystdlog, адрес которого будет отличаться от предыдущей реализации.

Поскольку использование инструментов является расширением языка, то включение механизмов с флагами [tool] требует снятия семантического ограничения на инструменты. Это делается оператором RESTRICT +TOOL.

Ассоциирование с данными

Технология программирования, управляемого данными, предполагает наличие данных, отделенных от программного кода. При этом возникает вопрос, каким инструментом обрабатывать каждый экземпляр данных. Должны быть использованы метаданные реализующей платформы. Что это означает для инструментального подхода?

Для структуры данных могут быть ассоциированы инструменты, для обработки данной структуры. Инструментов может быть несколько, на различных стадиях обработки используются разные инструменты.

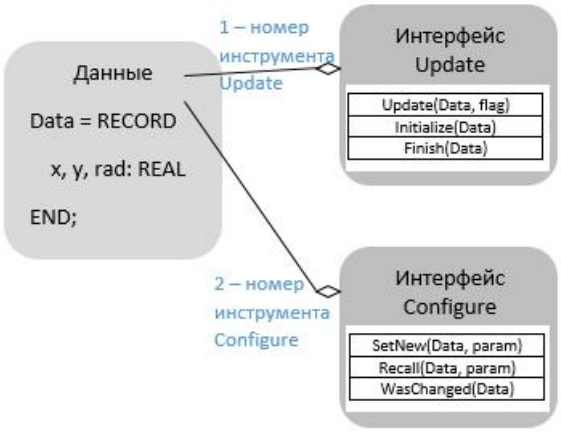


Рис. 4. Инструменты, ассоциированные с данными

Fig.4 Tools, associated with data

К метаданным реализующей платформы выдвигаются требования получения для каждого типа данных и номера инструмента указателя на инструмент. Если по данному номеру нет ассоциации, то, естественно, возвращается нулевой указатель.

Также следует добавить необходимость эффективного доступа к получению инструмента. Деревья и ассоциативные массивы для этих целей не подходят, требуется прямой индексный доступ на уровне информации в метаданных.

В МультиОбероне реализован эффективный индексный доступ к указателям на инструмент, рисунок 5.

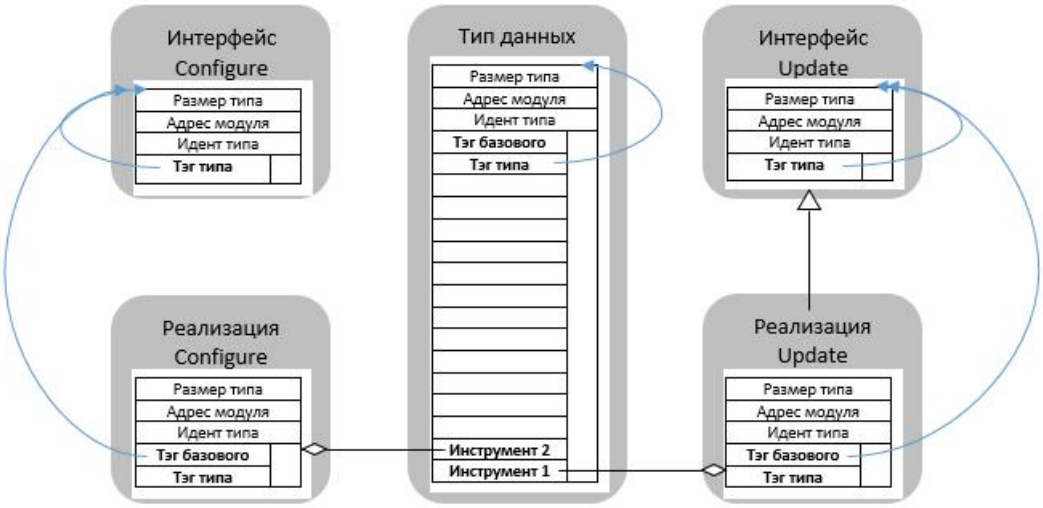


Рис. 5. Индексный доступ к инструментам

Fig.5 Indexed tool access

Для хранения указателей на тип инструмента используются свободные ячейки массива наследуемых типов base структуры Kernel.Type. Размер массива равен 16, что отражает максимальную глубину вложенности типов. При использовании инструментального подхода вложенные типы используются менее интенсивно. Для ассоциирования с первым инструментом используется ячейка 15, со вторым – 14 и т.д. Максимальное число ассоциированных инструментов определит компилятор.

Использование инструментов по данным

Для использования интерфейса Update по отношению к данным Data в ООП производится инсталляция объекта Data и вызов метода Update, который может быть виртуальным и подменяться реализациями для конкретных подтипов Data.

```
var Data data;

data.Update(flags);
```

Использование инструментального подхода предполагает наличие данных в персистентной области и запрос инструментов к ним.

```
VAR data: Data; tupdt: DataUpdate;

TOOL(tupdt, data, I_UPDT);

tupdt.Update(data, flags);
```

Системная функция TOOL с тремя аргументами предназначена для запроса инструмента, ассоциированного с данным типом по номеру интерфейса (в нашем случае, IFACE_UPDATE). Полученный инструмент применяется к данным вызовом метода Update с флагами.

Платой на разделение кода и данных является меньшая выразительность кода по сравнению с вариантом ООП.

Ассоциирование инструмента данным задается статически при определении типов.

```
CONST I_UPDT = 1;

Data = RECORD

x, y, rad: INTEGER

END;

DataUpdate = POINTER TO RECORD [tool] (IUpdate, Data, I_UPDT)

END;
```

Константа I_UPDT определяет номер инструмента. В определении инструмента закладывается, что инструмент ассоциирован с типом данных Data под номером I_UPDT.

Естественно, нет необходимости запроса TOOL(tupdt, data, I_UPDT) перед каждым вызовом метода. Это можно сделать один раз и далее использовать инструмент с относящимися к нему данными.

Выявленные проблемы и характеристики эргодичности

Проблема сохранения и восстановления сохраненных данных при сбоях и завершениях модулей программной системы не имеет простого решения для объектных архитектур. В сложных объектных системах сохранение требует сериализации, а восстановление - десериализации, что не всегда возможно выполнить при частичной неработоспособности распределенной системы. В результате такого восстановления возникает необходимость приведения объектов в требуемое состояние, что не всегда происходит. Это характеризуется внешними проявлениями, когда система работает, но не во всем наборе свойств.

В работе [7] рассмотрен ряд причин, приводящих к деградации свойств компьютерной системы со временем. Важным критерием является эргодичность, т.е. стабильность этих свойств, что требует доказательств, в том числе и архитектурного характера.

Использование данных в фиксированных персистентных хранилищах делает жизненный цикл изменения состояний системы более предсказуемым. Данные в таких хранилищах сбрасываются во временные файлы и восстанавливаются из них.

Вторая проблема восстановления связана с тем, что при таком восстановлении нужен механизм рестарта и приведения данных в соответствие. При разделении кода и данных при отказах в программных модулях кода нет никакой необходимости выгружать структуры данных из памяти. Модуль кода может перегружаться при необходимости, а данные остаются в памяти. Инструментальный подход гарантирует отсутствие подлежащих инициализации данных, т.к. инструменты, в отличие от объектов, не имеют полей данных.

Третьей проблемой является излишняя гибкость архитектуры системы реализующей платформы. Гибкость означает, что структуры данных и поведенческие характеристики объектной системы будут неопределенными, что не позволит оценивать изменение состояния компьютерной системы. В случае инструментального подхода структуры данных становятся более определенными, и более предсказуемой получается архитектура системы. Определенность структур данных означает, что состояние системы определяется этими структурами, и только ими. И, следовательно, стабильность этих структур данных приведет к улучшению характеристик эргодичности системы в целом.

Четвертой проблемой является возможность применимости решений для компактного встроенного ПО без использования указателей на объекты, размещенные в динамически выделяемой памяти. Указатель на инструмент является по определению единственным и не требует размещения экземпляра объекта в памяти.

Инструментальный подход и архитектурные решения, связанные с разделением кода и данных, приводят к более стабильным по характеристикам и доказуемым системам. Характеристики эргодичности предлагаемого подхода становятся лучше за счет явного состояния, хранимого в структурах данных. В конечном итоге динамическое поведение системы определяется состоянием и последовательностью входных воздействий, и, чем стабильнее начальное состояние, тем более предсказуемым окажется конечное состояние.

Другие способы решения и подходы

Объектные способы представления охватывают как представление данных, так и метаданных [8]. Более того, объектные модели, основанные на паттернах проектирования в настоящее время наиболее широко применяются для задач оси «знания-социальный мир» [9-11]. Наиболее важным отличием объектных подходов является то, что как данные, так и метаданные представляют собой объекты, вызываемые удаленно через адрес. При загрузках/восстановлениях кодов программ эти адреса, равно как и ссылки на них, изменяются и подлежат инициализации.

Время жизни модулей кода объектных систем будет равным времени жизни самой системы. Для решений, связанных с разделением кода и данных, время жизни системы будет равно времени жизни модулей данных и превышать время жизни модулей кода.

Важным механизмом повышения времени жизни систем является принцип

«ортогональной персистентности» [12], определяющий механизмы независимого сохранения состояния данных. Применение указанного механизма для Java-систем описывалось в [13]. Реальным большим шагом стала разработка ОС Phantom (Digital Zone, Россия), построенной на принципах ортогональной персистентности. Общим инструментального подхода и подхода на основе ортогональной персистентности является безударные механизмы сохранения и восстановления состояния. Но есть важные различия. Инструментальный подход позволяет реализовывать системы с восстановлением как на основе стандартных ОС, так и для критически важных систем, где ОС не рекомендуются или прямо запрещены [14]. Инструментальный подход не требует использования сборщика мусора, который «является серьезной проблемой для операционных систем» [15]. Микромир и системы реального времени накладывают еще большие ограничения на использования таких механизмов, как сборка мусора. Применение инструментального подхода более соответствует оси «реализация-детерминированный мир» и является предпочтительным механизмом для указанной области.

Применение в ПО СКАДА-платформы

Инструментальный подход использовался при разработке ПО СКАДИ (информация по https://scadi.tech), расшифровываемой как средства комплексной автоматизации и диагностики. Распределенная архитектура СКАДИ состоит из узлов, связанных между собой сетевыми протоколами. Каждый узел представляет собой адресуемый по сети компьютер с установленным ПО СКАДИ.

Обработчики осуществляют доступ к данным общей памяти, включая таблицы и очереди сообщений. Средства синхронизации ПО СКАДИ обеспечивают идентичность данных на любом узле с проверками и выдачей диагностики, если синхронизация на узлах не обеспечена.

Общая схема обновления данных по очередям сообщений представлена на рисунке 6.

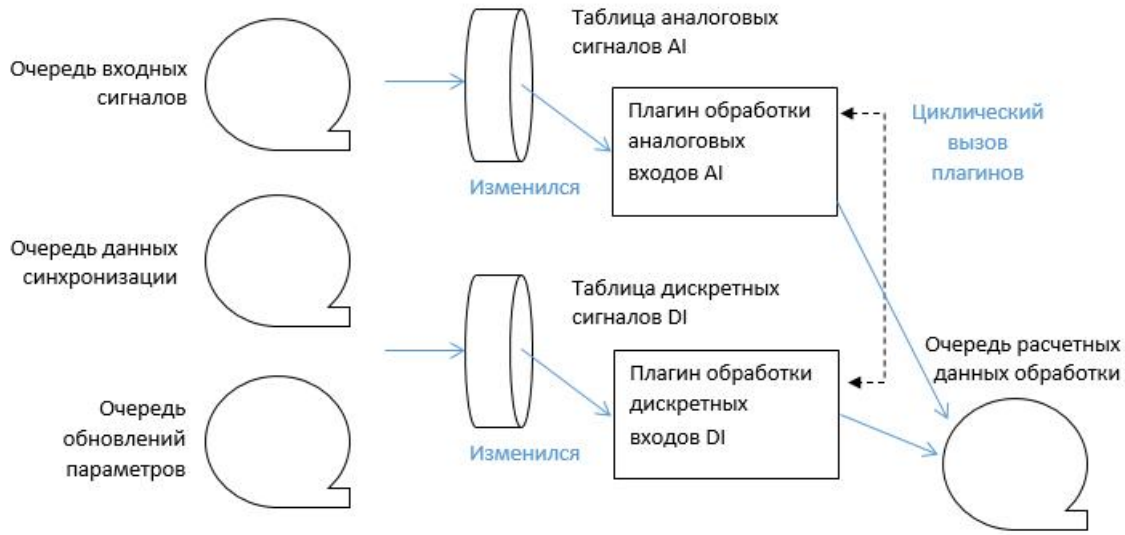


Рис. 6. Обработка данных СКАДИ

Fig.6 SCADI data processing

Программные сервисы DsDVP осуществляют обновление табличных данных узлов по

входным значениям очередей сообщений. Программа DsDVP каждому типу данных в общей памяти ассоциирует плагин обработки. Плагины обработки содержат исполняемые коды инструментов, соответствующих указанным типам данных. При переключении состояний задействованы автоматные механизмы [16], приведенные к виду инструмент – структура данных.

Например, плагин обработки для аналоговых данных AI содержит инструменты OvhAI, а плагин обработки дискретных данных DI – OvhDI. При этом объект обработки аналоговых данных AI содержит данные без методов, в то время как инструмент OvhAI обработки аналоговых данных содержит методы без данных.

Заключение

Предложенный в данной статье инструментальный подход позволяет сохранять преимущества классического структурного и объектно-ориентированного программирования. При этом инструментальный подход работает в инфраструктуре программирования управляемого данными. От объектно-ориентированного подхода берется полиморфизм и возможность работы по интерфейсам. От классического структурного программирования берется определение структур данных и взаимодействие с ними. От программирования, управляемого данными используется сепарация кода от данных и жизненный цикл последних в персистентном виде.

Сохранение и восстановление данных позволяет строить на базе инструментального подхода программные системы, у которых данные располагаются в специализированных, отдельно структурированных областях. Сохранение и восстановление таких данных не требует операций сериализации/десериализации и может осуществляться атомарно. Это особенно важно при построении высоконадежных систем 24x7, устойчивых к отказам в коде программных модулей. В свою очередь, программные коды сосредотачиваются в инструментах, которые не имеют состояния и, следовательно, не должны это состояние сохранять и восстанавливать.

Инструменты представляют собой программные объекты без данных, которые находятся в единственном экземпляре для каждого типа. Это позволяет отказаться от использования динамически выделяемой памяти на объекты, т.к. отсутствие данных не требует отсутствие экземпляра в памяти. Вместе с тем инструменты могут вызываться и обновлять ассоциированные с ними объекты – структуры данных.

Недостатком инструментального подхода является меньшая выразительность кода по сравнению с кодом ООП. В семантике вызова инструмента должны присутствовать как сам инструмент, так и ассоциированный экземпляр данных.

Инструментальный подход реализован в системе МультиОберон. Ограничительная семантика МультиОберона позволяет отключать функциональность ООП и включать функциональность инструментального подхода.

Подход «от ограничений» предполагает декларирование разработчиками того факта, что используется инструментальный подход, а значит все состояния данных сохраняются в персистентной памяти. Это декларации могут быть обусловлены стремлением повысить качество разрабатываемой системы или требованиями проектирования.

Использование инструментов и персистентных хранилищ данных существенно облегчает рассмотрение вопросов эргодичности. Такое разрабатываемое ПО будет меньше деградировать со временем.

Построение СКАДИ – системы с требованиями повышенной надежности реализовано с использованием инструментального подхода. Такое решение позволило позволило совершить еще один шаг по повышению качества программного продукта для построения ПО критически важных систем.

Библиография

1. David West. Object Thinking. Microsoft Press, 2004. С. 87-89.
2. Вирт Н., Алгоритмы и структуры данных. ДМК-Пресс, 2016. С. 272.
3. Н. Вирт, Ю. Гуткнехт. Разработка операционной системы и компилятора. Проект Оберон. ДМК-Пресс, 2017. С. 560.
4. Дагаев Д.В. Ограничительная семантика языка в системе МультиОберон // Программные системы и вычислительные методы. – 2023. – № 1. – С. 26-41. DOI: 10.7256/2454-0714.2023.1.36217 EDN: IWIODR
5. Rajive J., Ph.D. Data-Oriented Architecture: A Loosely-Coupled Real-Time SOA, Real-Time Innovations, Inc., 2007 August. С. 19-23.
6. Гамма Э., Хэлм Р., Джонсон Р., Влиссидес Дж. Приемы объектно-ориентированного проектирования. Паттерны проектирования. СПб: Питер, 2019. С. 368.
7. Дагаев Д.В. О разработке Оберон-системы с заданными свойствами эргодичности. Труды ИСП РАН, том 32, вып. 6, 2020 г., стр. 67-78. DOI: 10.15514/ISPRAS–2020–32(6)
8. Копылов М.С., Дерябин Н.Б., Денисов Е.Ю. Объектно-ориентированный подход к поддержке сценариев в системах оптического моделирования // Труды Института системного программирования РАН. 2023. No 35(2). стр. 169-180.
9. Nazar N., Aleti A., Zheng Y. Feature-based software design pattern detection // Journal of Systems and Software, 2022, vol. 185, pp. 1-12.
10. Yu D., Zhang P., Yang J., Chen Z., Liu C., Chen J. Efficiently detecting structural design pattern instances based on ordered sequences // Journal of Systems and Software, 2018, vol. 142, pp. 35–56.
11. Lo S. K., Lu Q., Zhu L., Paik H.-Y., Xu X., Wang C. Architectural patterns for the design of federated learning systems // Journal of Systems and Software, 2022, vol. 191, p. 357.
12. Hosking A., Nystrom N., Cutts Q., Brahnmath K. Optimizing the read and write barriers for orthogonal persistence // Advances in Persistent Object Systems, Morrison, Jordan, and Atkinson (Eds.). Morgan Kaufmann, 1999. p. 11.
13. Lefort A. A Support for Persistent Memory in Java // Computer science. Institut Polytechnique de Paris, 2023. English. p. 10.
14. ГОСТ Р МЭК 60880, Программное обеспечение компьютерных систем, выполняющих функции категории А // 2009. стр. 220.
15. Таненбаум А. Современные операционные системы // 4-е изд. – СПб.: Питер, 2015. Стр. 100-101.
16. Дагаев Д.В. Исполняющая машина автоматных программ // Научно-технический вестник информационных технологий, механики и оптики. 2021. Т. 21, № 4. С. 525-534.

Результаты процедуры рецензирования статьи

В связи с политикой двойного слепого рецензирования личность рецензента не раскрывается.

Со списком рецензентов издательства можно ознакомиться здесь.

Предмет исследования. Статья должна быть посвящена, исходя из названия, инструментальному подходу к программированию в системе МультиОберон. В целом, статья соответствует заявленной теме, однако отдельные моменты требуют доработки, о чём будет отмечено в соответствующих блоках рецензии.

Методология исследования автором выстроена на анализе и синтезе различных данных о предмете исследования. Ценно, что автор использует графический инструментарий представления полученных результатов. При этом, на представленных рисунках следует представить все надписи на русском языке (диссонирует основной текст статьи на русском языке и англоязычные рисунки). Также было бы хорошо увеличить разрешение подготовленных рисунков, т.к. внизу первого рисунка надписи на круге не видны.

Актуальность исследования вопросов, связанных с организацией программирования не вызывает сомнения, т.к. это напрямую связано с обеспечением технологического суверенитета Российской Федерации и вкладом в достижение цифровой трансформации (как одной из национальных целей развития Российской Федерации на период до 2030 года).

Научная новизна в представленном на рецензирование материале присутствует. В частности, связана с изложенными на рисунке 1 «осями знания-социальный мир и реализация-детерминированный мир». Более того, интерес представляют обозначенные семантические ограничения МультиОберона, что может быть использовано как в других научных исследованиях, так и в практической деятельности.

Стиль, структура, содержание. Стиль изложения научный. Структура статьи автором выстроена, способствует раскрытию темы, но её было бы интересно добавить блоком по решению выявленных проблем. Ознакомление с содержанием статьи не позволило чётко сформировать представление о системе выявленных автором проблем и путях их решения. При доработке статьи следует уделить внимание этому вопросу, т.к. это значительно расширит востребованность научной статьи.

Библиография. Автором сформирован достаточно скудный библиографический список, состоящий из 7 источников. В то же время ценно, что автор использовал как отечественные, так и зарубежные научные публикации. При проведении доработки статьи рекомендуется расширить перечень источников хотя бы до 15-20, прежде всего, за счёт изучения современной научной мысли (2022-2024 гг.), т.к. сейчас основной фокус внимания автор обращён к публикациям, вышедшим ранее 2020 года.

Апелляция к оппонентам. К сожалению, в тексте рецензируемой статьи автор не вступает в научные дискуссии с другими авторами, что снижает впечатление от ознакомления с данным материалом. При доработке рекомендуется устранить данное замечание, т.к. это позволит усилить глубину аргументации авторских суждений, а также покажет прирост научного знания по сравнению с тем, что уже имеется в научной литературе.

Выводы, интерес читательской аудитории. С учётом всего вышеизложенного заключаем о том, что статья может быть опубликована после проведения доработки по указанным в тексте рецензии замечаниям. Представляется, что статья на данную тему будет представлять большой интерес читательской аудитории журнала «Программные системы

и вычислительные методы».

Результаты процедуры повторного рецензирования статьи

В связи с политикой двойного слепого рецензирования личность рецензента не раскрывается.
Со списком рецензентов издательства можно ознакомиться здесь.

Рецензируемая статья посвящена разработке инструментального подхода к программированию на основе изучения преимуществ классического структурного и объектно-ориентированного программирования.
Методология исследования обобщении опыта классического структурного и объектно-ориентированного программирования, анализе научных публикаций и стандартов по изучаемой теме.
Актуальность работы обусловлена необходимостью интеграции преимуществ структурного и объектно-ориентированного программирования.
Научная новизна рецензируемого исследования по мнению рецензента, состоит в предложенном инструментальном подходе, при котором вместо интерфейса объекта, содержащего данные неизвестной структуры, используется интерфейс инструмента, не содержащего в себе данных и связанного с экземпляром данных, при этом инструмент, в отличии от объекта, не аллокируется в динамической памяти, не требует сериализации/десериализации, не может потерять данных при аварийном завершении из-за исключительной ситуации.
В статье структурно выделены следующие разделы: Введение, Объектное мышление против структурного программирования, Программирование, управляемое данными, Семантические ограничения МультиОберона, Инструментальный подход вместо объектного, Получение инструмента системной функцией TOOL, Расширение базового типа инструмента, Ассоциирование с данными, Использование инструментов по данным, Выявленные проблемы и характеристики эргодичности, Применение в ПО СКАДА-платформы, Заключение, Библиография.
В статье рассмотрены преимущества и области эффективного использования объектно-ориентированного программирования; применение программ, управляемых данными для построения слабосвязанных систем, использующих механизмы восстановления; освещены особенности языка модульного программирования Оберон; предложен инструментальный подход, заключающийся в создании вместо объекта инструмента, не содержащего в себе данные; отражены причины, приводящие к деградации свойств компьютерной системы со временем; применение средств комплексной автоматизации и диагностики. От объектно-ориентированного подхода автором берется полиморфизм и возможность работы по интерфейсам, от классического структурного программирования – определение структур данных и взаимодействие с ними; от программирования, управляемого данными – сепарация кода от данных.
Библиографический список включает 16 источников – публикации отечественных и зарубежных ученых по теме статьи, а также интернет-ресурсы и ГОСТы, на которые в тексте имеются адресные ссылки, подтверждающие наличие апелляции к оппонентам.
К недостаткам представленного на рецензирование материала можно отнести использование аббревиатур без расшифровок (например, ООП), а также оформление таблиц с отступлением от принятых правил – наименование таблицы 1 отражено после нее, а не перед таблицей как это принято ГОСТ.
Статья отражает результаты проведенного авторами исследования, соответствует направлению журнала «Программные системы и вычислительные методы», содержит элементы научной новизны и практической значимости, может вызвать интерес у

читателей, рекомендуется к опубликованию после внесения корректив в соответствии с высказанными замечаниями.

Программные системы и вычислительные методы
Правильная ссылка на статью:
Бондаренко В.А., Попов Д.И. Исследование и разработка алгоритмов к формированию эффективного ансамбля сверточных нейронных сетей для классификации изображений // Программные системы и вычислительные методы. 2024. № 1. DOI: 10.7256/2454-0714.2024.1.69919 EDN: WZDHQO URL: https://nbpublish.com/library_read_article.php?id=69919

Исследование и разработка алгоритмов к формированию эффективного ансамбля сверточных нейронных сетей для классификации изображений

Бондаренко Валерий Александрович

магистр, кафедра информационных технологий и математики, Сочинский государственный университет

354002, Россия, Краснодарский край, г. Сочи, ул. Верхняя Лысая Гора, 64

valeriybbond@mail.ru

Попов Дмитрий Иванович

доктор технических наук

профессор, кафедра информационных технологий и математики, Сочинский государственный университет

354002, Россия, Краснодарский край, г. Сочи, ул. Политехническая, 7

damitry@mail.ru

Статья из рубрики "Базы знаний, интеллектуальные системы, экспертные системы, системы поддержки принятия решений"

DOI:

10.7256/2454-0714.2024.1.69919

EDN:

WZDHQO

Дата направления статьи в редакцию:

20-02-2024

Аннотация: Объектом исследования являются искусственные нейронные сети (ИНС) со сверточной архитектурой для классификации изображений. Предметом исследования является исследование и разработка алгоритмов построения ансамблей сверточных нейронных сетей (СНС) в условиях ограниченности обучающей выборки. Целью

исследования является выработка алгоритма к формированию эффективной модели на основе ансамбля сверточных СНС с применением методов усреднения результатов каждой модели, способную избежать переобучения в процессе повышения точности прогноза и обученную на небольшом объеме данных, меньше 10 тысяч примеров. В качестве базовой сети в составе ансамбля выработана эффективная архитектура СНС, которая показала хороший результат в качестве одиночной модели. Также в статье исследованы методы объединения результатов моделей ансамбля и даны рекомендации к формированию архитектуры СНС. В качестве методов исследования используется теория нейронных сетей, теория машинного обучения, искусственного интеллекта, использованы методы алгоритмизации и программирования моделей машинного обучения, использован сравнительный анализ моделей, построенных на разных алгоритмах с использованием классического ансамблирования с простым усреднением и объединения результатов базовых алгоритмов в условиях ограниченности выборки с учетом средневзвешенного усреднения. Областью применения полученного алгоритма и модели является медицинская диагностика в лечебных учреждениях, санаториях при проведении первичного диагностического приема, на примере исследовательской задачи модель обучена классификации дерматологических заболеваний по входным фотоснимкам. Новизна исследования заключается в разработке эффективного алгоритма и модели классификации изображений на основе ансамбля сверточных НС, превосходящих точность прогноза базовых классификаторов, исследован процесс переобучения ансамбля классификаторов с глубокой архитектурой на малом объеме выборки из чего сделаны выводы по проектированию оптимальной архитектуры сети и выбору методов объединения результатов нескольких базовых классификаторов. По итогу исследования разработан алгоритм к формированию ансамбля СНС на основе эффективной базовой архитектуры и средневзвешенного усреднения результатов каждой модели для классификационной задачи по распознаванию изображений в условиях ограниченности выборки.

Ключевые слова:

искусственные нейронные сети, сверточная архитектура, ансамбли нейронных сетей, методы усреднения результатов, задача классификации, медицинская диагностика, ансамблирование нейронных сетей, средневзвешенное усреднение, предварительная обработка, взвешенное голосование

Введение

Для повышения точности прогноза и уменьшения ошибок можно использовать ансамбли моделей нейронных сетей [1]. Ансамбль – это комбинация нескольких моделей, работающих параллельно или последовательно и создающих объединенное предсказание на основе результатов каждой отдельной модели. Ансамбли алгоритмов машинного обучения представляют собой мощную технику, которая позволяет повысить точность и надежность классификации данных. Основная идея ансамблирования заключается в объединении нескольких базовых классификаторов таким образом, чтобы их индивидуальные ошибки компенсировались, а общая производительность системы улучшалась [2]. Существует несколько различных подходов к ансамблированию, каждый из которых имеет свои преимущества и недостатки. Как правило, ансамблированные классификаторы превосходят отдельные базовые классификаторы по точности и надежности. Ансамблирование алгоритмов машинного обучения находит широкое

применение в различных областях, таких как распознавание образов, обработка естественного языка, биоинформатика и финансовый анализ. Однако ансамблирование имеет некоторые недостатки, например, увеличение вычислительной сложности, потенциально более высокая склонность к переобучению, при недостаточной выборке, сложность выбора подходящего метода ансамблирования. В рамках исследования решается задача многоклассовой одновариантной классификации изображений, обучение модели относится к обучению с учителем и осуществляется на основе размеченного набора данных [3]. В ходе исследования мы сравним результаты нескольких моделей, первый классификатор построен на базе простого усреднения результатов от нескольких моделей ансамбля CHC с применением глубоких предварительно обученный сетей, второй классификатор будет разработан на основе одной эффективной архитектуры сети, на основе которой будет реализовано несколько моделей ансамбля, обученных на независимых выборках с применением средневзвешенного объединения результатов моделей.

Материалы и методы

Основной проблемой в использовании глубоких нейронных сетей на малом объеме данных является переобучение, это такое явление когда сеть очень хорошо прогнозирует на обучающей выборке и плохо на тестовой. Основными показателями переобучения сети является очень большая разница показателя качества на обучающей и валидационной выборке [3], также признаком переобучения является неизменность показателя качества от эпохи к эпохе рисунок 1.

Рисунок 1. Стагнация показателя точности при обучении

Мы видим, что начиная с 30 эпохи обучения изменение показателя практически не происходит, в этом случае происходит малое изменение весовых коэффициентов, это значит, что мы находимся в области малых градиентов или достигли локального минимума [5]. Также причиной переобучения является неоднородность при распределнии данных на тестовую и обучающую выборку, один класс получил нормальное соотношение отрицательного к положительному результату, другой класс нет, и имеет не репрезентативные данные, в результате при обучении сеть учится распознавать первый класс, второй нет, при усреднении результата по всем классам ухудшается качество итогового классификатора. Для предотвращения переобучения используют раннюю остановку обучения, ансамбли, методы оптимизации градиентного спуска, нормализацию, использование прореживающих слоев dropout, а также увеличение объема обучающей выборки [6].

Анализ результатов различных экспериментов показывает, что использование ансамблей моделей СНС может значительно увеличить точность прогнозирования. Это происходит благодаря тому, что каждая модель может обнаруживать определенные характеристики или паттерны в данных, которые могут быть упущены другими моделями [7]. Комбинирование предсказаний всех моделей позволяет снизить вероятность ложных прогнозов и повысить общую точность.

Более того, использование ансамблевых методов может помочь минимизировать эффект переобучения – явления, при котором модель "запоминает" данные обучающего набора и теряет способность к обобщению на новые данные [8]. Ансамблирование позволяет учитывать различные тренды в данных и создавать более устойчивую модель прогнозирования. Принцип работы ансамбля заключается в том, что каждая нейронная сеть обрабатывает входные данные независимо и выдает свой собственный прогноз. Затем, путем агрегации результатов всех нейронных сетей, получается конечный прогноз ансамбля.

На сегодняшний день существует множество методов ансамблирования моделей в зависимости от типа задачи. Исследуемая задача относится к многоклассовой одновариантной классификации изображений. Для распознавания образов применяются нейронные сети со сверточной архитектурой.

Объединение прогнозов из нескольких классификаторов является широко используемой стратегией для повышения точности классификации. Самый простой способ объединения прогнозов - это просто взять среднее арифметическое. Однако этот подход может быть неэффективен, если отдельные классификаторы имеют сильно различающуюся точность [9]. Более эффективным способом объединения прогнозов является использование взвешенного среднего, где каждому классификатору присваивается вес в соответствии с его точностью. Вес классификатора может быть определен на основе его производительности на проверочном наборе данных. Для поиска оптимальных весов можно использовать различные методы оптимизации, такие как случайный поиск, дифференциальный поиск или оптимизация Нелдера-Мида [10]. Оптимальные веса могут зависеть от конкретного набора данных и используемых классификаторов. Другим важным фактором, влияющим на эффективность ансамбля классификаторов, является разнообразие классификаторов. Классификаторы, которые делают схожие ошибки, не будут дополнять друг друга и не улучшат точность ансамбля. Поэтому желательно использовать классификаторы, которые используют разные методы классификации или имеют разные архитектуры [9].

Ввиду того, что основная часть алгоритмов для классификации основывается на поиске экстремума определенной целевой функции, возможно, что классификационное решение окажется вблизи локального экстремума. Чтобы повысить вероятность нахождения глобально-оптимального решения, можно использовать ансамбли моделей, которые объединяют результаты различных базовых классификаторов, построенных на разных выборках исходных данных. Такой подход позволяет искать решение из разных точек, что способствует повышению шансов на получение оптимального результата [11].

Ансамбль базовых алгоритмов вида $a_w(z) = f(b_w(z))\ (w = 1,..,W)$ может принимать вид: $a(z) = f(corr(b_1(z), b_2(z),...,b_W(z)))$. То есть, это последовательность операций: bw: K->H, операции корректировки corr: Hw->H и решающие правила f: H->U, где K-объекты данных, U-множество размеченных классов, H и U это набор данных, включающий объекты или кортежи данных и соответствующие им метки классов, H это множество оценок [11].

Тогда, например, если у нас имеется 5 независимых алгоритмов, каждый из которых определяет класс изображения с некой вероятностью p0, то вероятность после ансамблирования будет равна $p_0^5 + C_5^4 p_0^4 (1-p) + C_5^3 p_0^3 (1-p)^2$.

Существует несколько способов объединения результата в ансамбле [12]:

-простое голосование:

$$a(\vec{x}) = \frac{1}{T}\sum_{t=1}^{T} a_t(\vec{x});$$

(1)

-взвешенное голосование:

$$a(\vec{x}) = \sum_{t=1}^{T} \alpha_t a_t(\vec{x}),$$

(2)

где at можно подбирать линейными методами, $\alpha_t > 0; \sum \alpha_t = 1$;

-смешенное голосование:

$$a(\vec{x}) = \sum_{t=1}^{T} g_t(\vec{x}) a_t(\vec{x}, \overrightarrow{\theta_t}),$$

(3)

где gt(x) это функция компетентности, которая способна определить обрасти где более уместен тот или иной алгоритм.

При простом голосовании, выбирается класс, который наиболее соответствует большинству голосов базовых классификаторов. Однако взвешенное голосование различается от простого голосования в том, что оно присваивает веса голосам базовых классификаторов, учитывая их точность работы. Возможно использование функциональной зависимости как коэффициента веса при сочетании экспертных мнений. Помимо различных методов голосования, смешивание результатов ансамбля может быть выполнено через усреднение, как с учетом веса, так и без него. Например, если результат классификации представлен вероятностями принадлежности объектов классам, а не метками классов. При усреднении без учета веса, итоговое значение ансамбля рассчитывается как простое среднее всех результатов базовых классификаторов. В случае взвешенного усреднения, результаты, полученные от базовых классификаторов, умножаются на соответствующие весовые коэффициенты [11].

Для оптимизации ансамблей нейронный сетей часто применяют метод бустинга (стохастический, адаптивный, градиентный), они заложены в методах адаптивной оптимизации НС, таких как Adagrad, rmsprop, adadelta, метод адаптивной оценки моментов (Adam) [13, 14, 15].

Преимущества использования ансамблей сетей: имеют большую гибкость, возможность достичь более лучших показателей качества моделей за счет вариативности различных алгоритмов, однако использование ансамблей является затратным в плане вычислительных мощностей [11].

Широкое применение ансамблей нейронных сетей возникает в ситуациях, где имеется огромный объем данных, больше 100 тысяч примеров. В данном случае, набор нейронных сетей образует ансамбль, превосходящий случайный выбор в предсказательной точности при данной плотности данных. Эти сети имеют простую структуру, что позволяет быстро обучаться. Их теоретическое основание основано на принципе центральной предельной теоремы в теории вероятностей. Эта теорема утверждает, что последовательность средних значений, вычисленных на основе независимого набора из корня из n случайных величин, имеющих большую дисперсию σ и подчиняющихся нормальному распределению, сходится к нормальному распределению. Ансамбль нейронных сетей использует этот принцип для повышения качества прогнозов [16]:

$$P(\mathrm{O}_i|\Pi_j) = \frac{P(\Pi_j|\mathrm{O}_i)P(\mathrm{O})}{P(\Pi)},$$

(4)

где, $P(\mathrm{O}_i|\Pi_j)$ - это условная вероятность отрицательной погрешности i-ой сети на t-ом шаге обучения при положительной погрешности j-ой сети, $P(\Pi_j|\mathrm{O}_i)$ - это условная вероятность положительной погрешности j-ой сети при отрицательной i-ой, P(П), P(O) - сами вероятности появления положительного и отрацительного прогноза.

Путем объединения прогнозов нескольких нейронных сетей и их усреднения можно значительно снизить среднеквадратическое отклонение на √n. Иначе говоря, если суммировать прогнозы отдельных моделей при определенных условиях, то можно уменьшить неопределенность, сопутствующую каждой из них. Таким образом, точность прогнозирования возрастает с увеличением количества использованных нейронных сетей и среднего значения их выходных сигналов. Важно, чтобы ошибки прогнозов отдельных нейронных сетей были статистически независимыми, для достижения этого можно применить различные методы, например, обучение нейронных сетей на разных наборах данных и использование разнообразных моделей [16].

Для достижения высокого качества ансамбля нейронных сетей необходимо обеспечить их полную статистическую независимость. Это означает, что каждая сеть должна обучаться на различных и независимых наборах данных. Однако, усреднение результатов от этих сетей может не привести к существенному снижению дисперсии выходного сигнала. Поэтому в таких случаях часто используется метод взвешенного голосования. Путем присвоения различных весов каждой сети, можно достичь оптимального результата. Этот подход позволяет учесть вклад каждой сети и создать более точный и надежный ансамбль [16-20]:

$$\hat{y}_R(\hat{\mathbf{y}}) = \frac{1}{n}\sum_{i=1}^{n} a_i \hat{y}_i(\mathbf{x}) + b,$$

(5)

где yi(x) это выход нейронной сети, yr(y) это вероятность принадлежности к классу, y это вектор вероятностей принадлежности к классами, b это весовой коэффициент.

В этом случае дисперсия будет определяться по формуле:

$$\sigma_R^2 = \frac{1}{n}\left(\sum_{i=1}^{n} a_i^2 \sigma_i^2 + 2\sum_{i<j} a_i a_j r_{ij} \sigma_i \sigma_j\right),$$

(6)

где rij это коэффициент корреляции между выходами от нескольких нейронных сетей.

Существует необходимость в использовании многоуровневой аппроксимации для уменьшения дисперсии результирующего прогноза на основе коррелирующих выходных сигналов i-ой и j-ой нейронных сетей на реальных наборах данных. Для этого важно создать многоуровневую архитектуру ансамбля с 2-3 уровнями, где каждый уровень будет аппроксимировать прогнозы предыдущего уровня. Дополнительным условием является минимальная корреляция выходных сигналов нижнего уровня. Веса при этом задаются самостоятельно и с помощью специальных методов осуществляется поиск оптимальных весов, показывающих вклад каждой модели в итоговый прогноз [16].

Для проведения экспериментов по классификации изображений был выбран открытый источник данных по дерматологическим заболеваниям ISIC HAM10000[21]. Он содержит 10015 изображений заболеваний кожи. Помимо загрузки изображений, загружен файл csv с размеченными данным по изображениям, это возраст пациента, локализация образования, пол, код изображения и целевой класс это диагноз. Данный набор был выбран так как имеется полностью размеченная выборка, данные являются репрезентативными и является самым большим набором в этой области. Исследование велось на языке python с аппаратным графическим ускорителем, с использованием фреймворков машинного обучения и разработки нейронных сетей, таких как keras, tensorflow, sklearn.

Результаты и обсуждение

Для создания ансамбля сетей было принято решение обучить несколько сверточных нейронных сетей с различными параметрами и архитектурой на разных обучающих выборках. Мы считали наши изображения и преобразовали их в вектора тензоров с помощью библиотеки nampy, при этом изменив размерность входного изображения, чтобы снизить нагрузку на вычислительные мощности. Классы были очень не сбалансированы, поэтому мы провели очистку данных самого большого по объему класса и аугментировали данные по количеству максимального класса, путем вращений, изменения цвета, обрезки изображений, чтобы добиться сбалансированности целевых классов. Затем мы считали полученный датафрейм с разметками классов и объединили с полученными тензорами изображений по коду изображения.

Предварительная обработка данных также очень важна для повышения качества будущего классификатора, поэтому в ходе анализа, полученный набор данных был исследован на аномалии, преобразование категориальных переменных, трансформацию данных, удаление дубликатов [22, 23]. С помощью метода train_test_split мы разделили выборку на обучающую и тестовую, где 80% обучающей выборки, 20% на тестовую. По полученным данным провели нормализацию и стандартизацию данных. Затем полученную обучающую выборку мы также разделили на обучающую и валидационную в размере 90% обучающей и 10% валидационной. Полученную обучающую выборку мы также разделили на три независимых набора с вероятностью выбора 20% для каждой модели.

Для минимизации корреляции между прогнозами отдельных моделей мы их обучили на разных независимых выборках. Первая модель спроектирована согласно методу трансферного обучения [24], мы подгрузили архитектуры и веса нескольких предварительно обученных моделей на ImageNet, с разной архитектурой с целью минимизации дисперсии погрешностей, показавшие хорошие результаты в

распознавании образов, такие как InceptionV3, InceptionResNetV2, VGG16 от фреймворка keras и tensorflow [25, 26, 27]. Задали размерность входного изображения 75x75 пикселей ввиду ограниченности вычислительных мощностей, после чего мы заморозили слои предобученных моделей и изменили последний слой под количество целевых классов, в нашей задаче их 7 [28, 29]. На рисунке 2 показана схема реализации трансферного обучения.

```
from tensorflow.keras.applications import InceptionV3, InceptionResNetV2, VGG16
model1 = InceptionResNetV2(weights='imagenet', include_top=False, input_shape=(75, 75, 3))
model2 = InceptionV3(weights='imagenet', include_top=False, input_shape=(75, 75, 3))
model3 = VGG16(weights='imagenet', include_top=False, input_shape=(75, 75, 3))
model1.trainable = False
model2.trainable = False
model3.trainable = False
model1_v1 = model1.output
model1_v2=GlobalAveragePooling2D()(model1_v1)
model1_v3 = Dense(1024, activation='relu')(model1_v2)
pred1 = Dense(7, activation='softmax')(model1_v3)
model_final1 = Model(inputs=model1.input, outputs=pred1)
model2_v1 = model2.output
model2_v2=GlobalAveragePooling2D()(model2_v1)
model2_v3 = Dense(1024, activation='relu')(model2_v2)
pred2 = Dense(7, activation='softmax')(model2_v3)
model_final2 = Model(inputs=model2.input, outputs=pred2)
model3_v1 = model3.output
model3_v2=GlobalAveragePooling2D()(model3_v1)
model3_v3 = Dense(1024, activation='relu')(model3_v2)
pred3 = Dense(7, activation='softmax')(model3_v3)
model_final3 = Model(inputs=model3.input, outputs=pred3)
model_final1.compile(optimizer=Adam(), loss='categorical_crossentropy', metrics=['accuracy'])
model_final2.compile(optimizer=Adam(), loss='categorical_crossentropy', metrics=['accuracy'])
model_final3.compile(optimizer=Adam(), loss='categorical_crossentropy', metrics=['accuracy'])
```

Рисунок 2. Реализация трансферного обучения

Полученные модели мы скомпилировали, в качестве оптимизации мы выбрали оптимизатор Adam, в качестве функции потерь categorical_crossentropy, в качестве метрики выбрали accuracy. Каждую модель мы обучили на 40 эпохах и размером батча в 10 пакетов на разных выборках. Каждая модель показала разный результат, худший результат показала модель InceptionV3 в 53% точности, поскольку она имеет самую глубокую архитектуру и большее количество параметров, поэтому обучение на малом объеме данных показывает низкое качество. Для реализации объединения результатов простым голосованием мы применили слой Average для усреднения выходного сигнала от каждой модели. На рисунках 3, 4 показана реализация метода объединения результата.

```
models = [model_final1, model_final2, model_final3]
input = Input(shape=(75, 75, 3), name='input')
outputs = [model(input) for model in models]
x = Average()(outputs)
x = Dropout(0.5)(x)
output = Dense(7, activation='softmax', name='output')(x)
conc_model = Model(input, output, name= 'Model')
conc_model.compile(optimizer=Adam(), loss='categorical_crossentropy', metrics=['accuracy'])
```

Рисунок 3. Реализация простого усреднения предобученных моделей

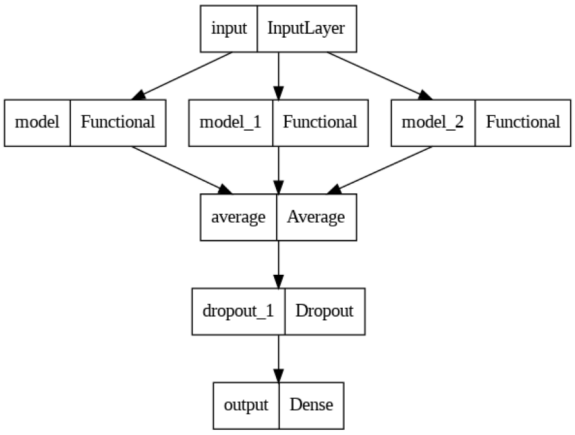


Рисунок 4. Архитектура ансамбля с простым усреднением

Мы создали новую архитектуру модели, добавили новый слой Input для входного изображения, от каждой модели получили выходные модернизированные слои и передали их в усредняющий слой Average. С целью предотвращения переобучения добавили прореживающий слой dropout и выходной полносвязный слой с 7 целевыми классами и функцией активации softmax. Полученная архитектура скомпилирована и обучена аналогично. На 50 эпохе обучения точность на обучающей выборке составляла 71%, точность на валидационной выборке 76%. На рисунке 5 показан процесс обучения. Точность на тестовой выборке составила 74% рисунок 6.

```
Epoch 45/50
1512/1512 [==============================] - 74s 49ms/step - loss: 0.8227 - accuracy: 0.7094 - val_loss: 0.7243 - val_accuracy: 0.7625
Epoch 46/50
1512/1512 [==============================] - 77s 51ms/step - loss: 0.8260 - accuracy: 0.7093 - val_loss: 0.7242 - val_accuracy: 0.7625
Epoch 47/50
1512/1512 [==============================] - 72s 48ms/step - loss: 0.8353 - accuracy: 0.7060 - val_loss: 0.7242 - val_accuracy: 0.7625
Epoch 48/50
1512/1512 [==============================] - 72s 48ms/step - loss: 0.8266 - accuracy: 0.7095 - val_loss: 0.7241 - val_accuracy: 0.7625
Epoch 49/50
1512/1512 [==============================] - 75s 50ms/step - loss: 0.8185 - accuracy: 0.7095 - val_loss: 0.7241 - val_accuracy: 0.7625
Epoch 50/50
1512/1512 [==============================] - 76s 50ms/step - loss: 0.8193 - accuracy: 0.7138 - val_loss: 0.7241 - val_accuracy: 0.7625
```

Рисунок 5. Процесс обучения ансамбля с простым усреднением

```
accuracy = accuracy_score(np.argmax(y_test2, axis=1), y_pred)
precision = precision_score(np.argmax(y_test2, axis=1), y_pred, average='macro')
recall = recall_score(np.argmax(y_test2, axis=1), y_pred, average='macro')
f1 = f1_score(np.argmax(y_test2, axis=1), y_pred, average='macro')
print(f"Accuracy: {accuracy:.4f}")
print(f"Precision: {precision:.4f}")
print(f"Recall: {recall:.4f}")
print(f"F1-score: {f1:.4f}")
```

```
Accuracy: 0.7412
Precision: 0.7411
Recall: 0.7440
F1-score: 0.7421
```

Рисунок 6. Оценка качества модели ансамбля на тестовой выборке

Однако мы видим, что на последних эпохах изменений в качестве обучения не происходило, что говорит о переобучении модели, поскольку обучающая выборка составляла меньше 10 тысяч примеров, что для обучения нейронных сетей с очень глубокой архитектурой является недостаточным, поэтому разработка собственной архитектуры с минимальным количеством уровней свертки является более оптимальным для этой задачи, причем количество классификаторов в ансамбле должно быть больше 3 и он должен иметь иерархическую архитектуру. Использование слабой архитектуры мы применим совместно с алгоритмом формирования средневзвешенного ансамбля [14]. В ходе экспериментальных исследований был выработан алгоритм к формированию ансамбля СНС.

Первым шагом мы спроектировали и оценили качество эффективной архитектуры CHC как базовой модели. На рисунке 7 представлена архитектура эффективной CHC.

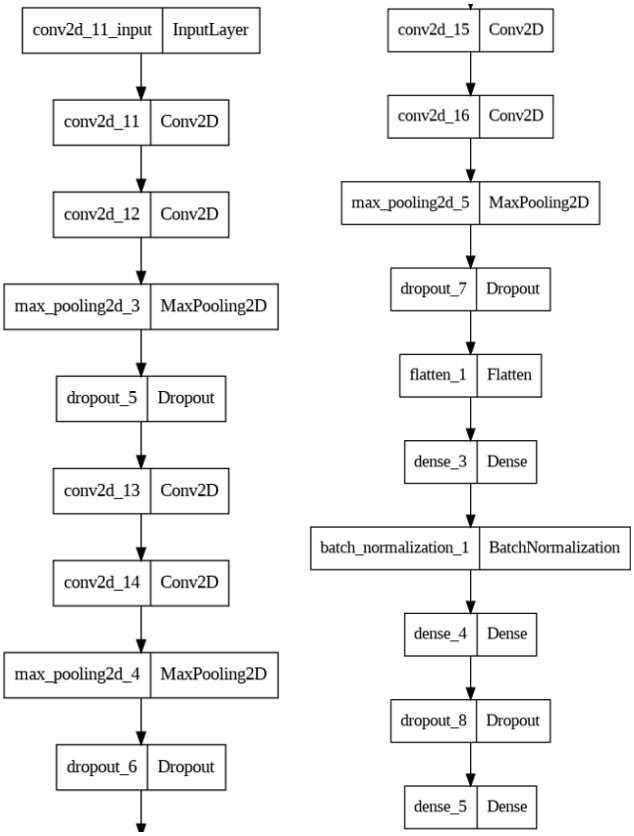


Рисунок 7. Эффективная архитектура CHC

Данная модель состоит из трех сверточных уровней, в каждом из которых имеется два сверточных слоя, слой пулинга и слой прореживания, экспериментальным путем подобраны гиперпараметры сети, количество фильтров в слое с увеличением на каждом следующем слое в 2 раза, размер ядра начиная с 7x7 и последующем применении ядра 3x3, функции активации, чередование слоев, в конце добавлен один полносвязный слой, слой нормализации flatten, прореживания dropout и один выходной полносвязный слой dense для классификации 7 целевых классов. В ходе исследования выработаны рекомендации к формированию архитектуры эффективного нейросетевого классификатора [30-33]:

-необходима предварительная обработка данных, анализ сбалансированности классов, в случае малого объема выборки необходимо обогащение;

-чем выше размерность входного изображения, тем большее количество уровней свертки необходимо использовать;

-также важен выбор оптимизатора при компиляции сети, если эпох обучения задано много, необходимо использовать оптимизатор SGD, иначе Adam;

-должно быть чередование слоев свертки и слоя пулинга, количество слоев свертки задается оптимально от 1 до 2 в зависимости от сложности входного изображения, для огромных объемов данных, больше 100 тысяч примеров, можно использовать количество сверток больше 3, количество слоей пулинга от 0 до 1 на одном уровне свертки с размером ядра (2,2) и функцией MaxPooling2D;

-на последних слоях необходимо уменьшать количество полносвязных слоев, оптимальным является один полносвязный слой и один выходной полносвязный слой;

-для уменьшения вероятности переобучения необходимо использовать слои dropout и BatchNormalization;

-важным параметром слоя свертки является размер ядра фильтра свертки, которое может принимать значения (3,3), (5,5), (7,7), чем сложнее входное изображение, тем выше необходимо задавать размер фильтра сверточного слоя;

-размер входного изображения input_shape на входном слое должен быть квадратным;

-параметры padding и stride сверточного слоя должны быть целочисленными и равны значениям предыдущего слоя, если он не был тоже сверточным;

-количество фильтров filters в сверточном слое задается оптимальным образом, чем выше размер ядра фильтра, тем большее число задается, и с каждым последующим уровнем свертки увеличивается в 2 раза.

В ходе исследования выявлены важные параметры архитектуры нейронной сети, которые влияют на ее эффективность в процессе обучения: размер фильтра сверточного слоя, тип чередования слоев сети, а также количество слоев свертки на одном уровне, количество фильтров, параметр filters, используемых в сверточном слое, количество уровней свертки.

Следующим шагом мы обучили модель, она показала достаточно хороший результат, практически 90% точности на обучающей и 78% на тестовой выборке рисунок 8. Данную архитектуру мы применили для ансамбля с оптимизированным поиском весов и средневзвешенным усреднением.

Рисунок 8. График изменения точности в процессе обучения эффективной СНС

Библиотека SciPy имеет большое количество методов оптимизации, например, метод differential_evolution [34], он находит глобальный минимум многомерной функции, при этом не использует градиентные методы и носит случайный характер, на вход функции подается функция для оценки набора весов, границы весов, количество моделей в ансамбле, тестовый набор X и y, количество итераций поиска, применяя данный метод можно минимизировать дисперсию выходного сигнала моделей ансамбля.

Реализована функция обучения моделей, в которой задана архитектура и для каждой модели формируется отдельный набор данных для обучения, чтобы обеспечить

независимость моделей ансамбля во время обучения. Обучение таких моделей происходит на разных выборках, сами архитектуры должны иметь различающиеся параметры, что повысит адаптивность к изменению входных данных ансамбля. Следующим шагом реализованы функции оценки качества ансамбля и набора весов [35] На рисунке 9 показаны основные функции алгоритма, реализованные на python.

```
def ensemble_predict(m, w, x_test4):
  y = [model.predict(x_test4) for model in m]
  y = array(y)
  s = tensordot(y, w, axes=(0,0))
  res = argmax(s, axis=1)
  return res
def evaluate_ensemble(m, w, x_test4, y_test2):
  y = ensemble_predict(m, w, x_test4)
  return accuracy_score(y_test2, y)
def normalize(w):
  res = norm(w, 1)
  if res == 0.0:
    return w
  return w / res
def loss_function(w, m, x_test4, y_test2):
  norm = normalize(w)
  return 1.0 - evaluate_ensemble(m, norm, x_test4, y_test2)
```

Рисунок 9. Реализация основных функций метода взвешенной оценки

Для реализации алгоритма поиска оптимальных весов необходима функция оптимизации, для этого была создана функция loss_function, функция оценки весов получает на вход вектор весов w, количество моделей ансамбля m, а также тестовые X_test и y_test, в результате функции полученный вес нормализуется с помощью второй функции normalize, в которой с помощью nampy.norm нормализуется значение веса, исходный вес делится на нормализованную величину, по итогу функции loss рассчитывается доля истинно отрицательных ответов и рассчитывается как 1 минус точность положительного прогноза, для этого вызывается функция evaluate_ensemble, в которую передается количество моделей, нормализованная величина веса и тестовый пример, в этой функции рассчитывается точность положительного прогноза с помощью accuracy_score, в которую передается фактический класс из тестового набора и прогнозируемый класс, для определения прогнозируемого класса вызывается функция ensemble_predictions, в которой для каждой модели делается прогноз на тестовом наборе с помощью метода predict, формируется массив вероятностей принадлежности к классам, затем с помощью метода tensordot формируется скалярное тензорное произведение по осям вероятностей на полученный вес, то есть на вход метод получает матрицу размером 5x7, 5 моделей ансамбля и 7 целевых классов, по каждому классу модель сформировала вероятность принадлежности, метод транспонирует матрицу, тогда по горизонтали получается 7 по вертикали 5, затем каждый вектор матрицы умножает на заданный вес, после чего суммирует взвешенные сигналы моделей по столбцам, тем самым получается 7 взвешенных вероятностей принадлежности от каждой модели ансамбля, из массива вероятностей принадлежности с учетом весов выбирается максимальное значение и берется его индекс с помощью метода argmax. В результате мы получаем прогнозируемый класс с учетом заданных весов.

Функция оценки весов в нашем случае является функцией потерь и определяет долю истинно отрицательных результатов, которые ухудшают качество и определяется как 1 минус точность положительного прогноза. На рисунке 10 показана блок-схема алгоритма

поиска оптимальных весов.

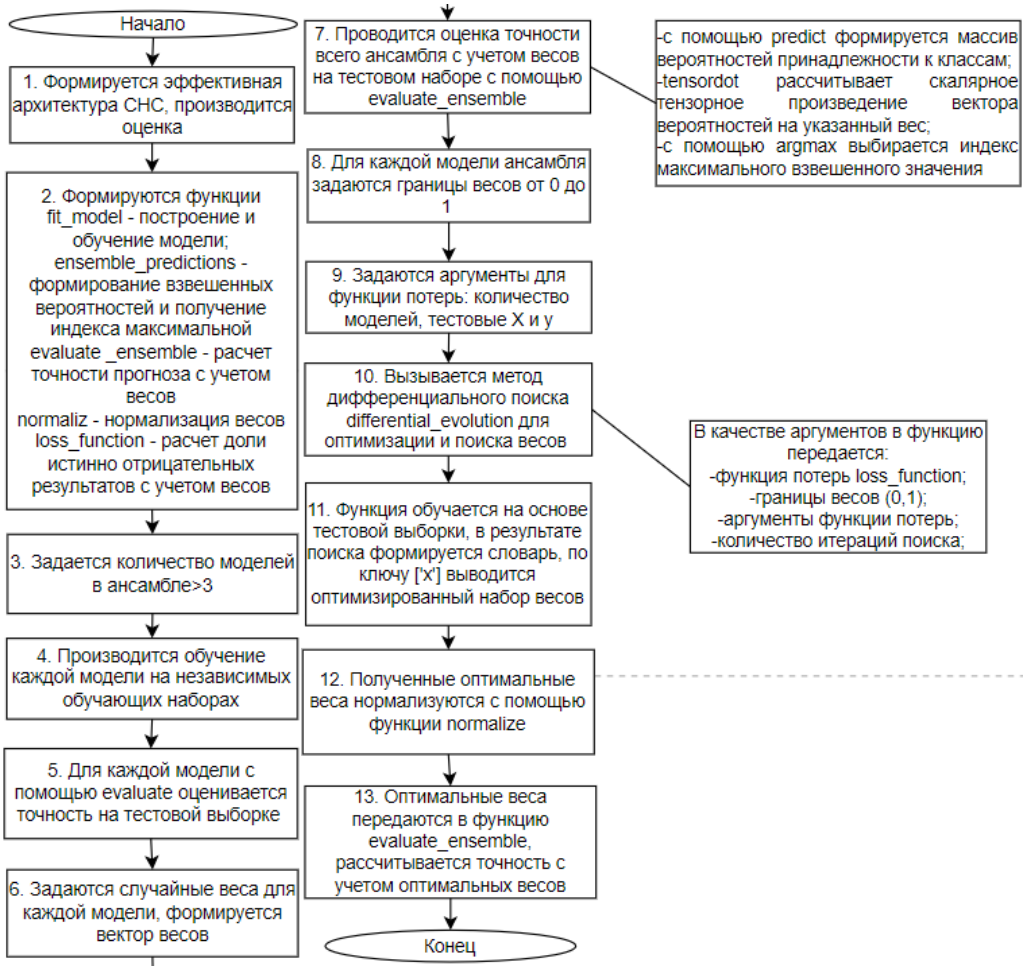


Рисунок 10. Блок-схема алгоритма формирования взвешенного ансамбля

Задается количество моделей в ансамбле, каждая модель обучается на своем независимом наборе данных, с помощью метода evalute каждая модель оценивается и рассчитывается ее точность. Формируется список весов для каждой модели, с помощью функции ensemble_predictions оценивается точность ансамбля с учетом заданных весов, веса на начальном этапе задаются случайными, задаются границы весов от 0 до 1, указываются аргументы функции оптимизации весов, это количество моделей, тестовый набор X и y, после чего вызывается функция оптимального поиска весов differential_evolution, в которую передается функция оптимизации, ограничение весов, аргументы и задается количество итераций поиска. По ключу X мы получаем оптимальный набор весов для каждой модели вида (0.01, 0.2, 0.11, 0.49, 0.12), далее оцениваем качество ансамбля по уже оптимальным весам. Мы апробировали результаты алгоритма на тестовом наборе данных. В результате алгоритм формирует 5 сверточных сетей и оценивает их точность таблица 1.

Модель	Точность
CNN 1	0,80
CNN 2	0,82
CNN 3	0,81
CNN 4	0,83
CNN 5	0,81

Таблица 1. Результаты взвешенных оценок моделей ансамбля

В таблице 2 представлено сравнение полученных оценок точности каждой модели.

Метод	Accuracy	Precision	Recall	F1 мера
Ансамбль из трех предварительной обученных моделей VGG16, InceptionV3, InceptionResNetV2 на основе простого усреднения результатов моделей	0,74	0,74	0,74	0,74
Одиночная СНС с эффективной архитектурой согласно алгоритму	0,78	0,78	0,78	0,77
Ансамбль из 5 эффективных СНС на основе взвешенной оценки результатов согласно алгоритму	0,83	-	-	-

Таблица 2. Оценка качества полученных моделей

Модель формирует результат точности с учетом взвешенных оценок, оценка точности на оптимальных весах равна 0,83, что выше, чем у модели на основе ансамбля глубоких предварительно обученных сетей. Алгоритм находит оптимальные веса и показывает весовую значимость каждой модели ансамбля, полученные веса передаются в функцию ensemble_predictions, тем самым прогнозируется точность с учетом оптимальных весов, на основе полученных весов формируется оценка точности объединяющего классификатора, которая достигает 83%, что на 5% выше чем, у одиночной эффективной архитектуры СНС и на 9% выше, чем у ансамбля из глубоких предобученных моделей с простым усреднением результатов.

Работу модели мы проверили на тестовом примере, мы подгрузили тестовое изображение дерматологического заболевания меланоцитарного невуса и дерматофибромы, уменьшили размерность к (75, 75), поскольку модель обучалась на этой размерности, затем преобразовали в тензорное представление с помощью библиотеки nampy.asarray, чтобы добиться 0 среднего значения и 1 дисперсии полученный вектор тензоров нормализовали путем вычета среднего и деления на стандартное отклонение. На рисунке 11 показана программная реализация Загрузки и обработки изображения.

```
f_name='/content/drive/MyDrive/Colab Notebooks/ISIC_0024319.jpg'
image=np.asarray(Image.open(f_name).resize((75,75)))
plt.imshow(image)
plt.show()
img=np.asarray(image.tolist())
x__mean = np.mean(img)
x__std = np.std(img)
img = (img - x__mean)/x__std
img=np.asarray(img)[None, ...]
img.shape
```

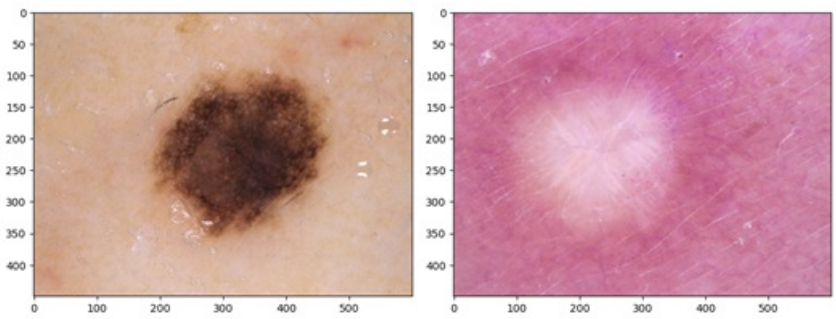


Рисунок 11. Загрузка и обработка входного изображения

Затем мы написали функционал на python для прогнозирования класса заболевания и отображения гистограммы распределения вероятностей по целевым классам. На рисунке

12 показан программная реализация для двух тестовых фотоснимков.

```
import matplotlib.pyplot as plt
lesion_type_idx_dict = {
    4: 'Melanocytic nevi',
    5: 'Melanoma',
    2: 'Benign keratosis-like lesions ',
    1: 'Basal cell carcinoma',
    0: 'Actinic keratoses',
    6: 'Vascular lesions',
    3: 'Dermatofibroma'}
print(conc_model.predict(img))
classes=np.argmax(conc_model.predict(img), axis=-1)
print(lesion_type_idx_dict.get(classes[0]))
pred=conc_model.predict(img)
bur=pred[0]
t= {i: bur[i] for i in range(len(bur))}
t=pd.DataFrame(list(t.items()), columns=['Key', 'Values'])
t['Description'] = t['Key'].map(lesion_type_idx_dict)
plt.barh(t['Description'],t['Values'], color='blue')
```

```
1/1 [==============================] - 0s 236ms/step
[[0.01158232 0.01232567 0.03795601 0.00943617 0.84017426 0.04994805
  0.03857748]]
1/1 [==============================] - 0s 292ms/step
Melanocytic nevi
```

```
import matplotlib.pyplot as plt
lesion_type_idx_dict = {
    4: 'Melanocytic nevi',
    5: 'Melanoma',
    2: 'Benign keratosis-like lesions ',
    1: 'Basal cell carcinoma',
    0: 'Actinic keratoses',
    6: 'Vascular lesions',
    3: 'Dermatofibroma'}
print(conc_model.predict(img))
classes=np.argmax(conc_model.predict(img), axis=-1)
print(lesion_type_idx_dict.get(classes[0]))
pred=conc_model.predict(img)
bur=pred[0]
t= {i: bur[i] for i in range(len(bur))}
t=pd.DataFrame(list(t.items()), columns=['Key', 'Values'])
t['Description'] = t['Key'].map(lesion_type_idx_dict)
plt.barh(t['Description'],t['Values'], color='blue')
```

```
1/1 [==============================] - 0s 435ms/step
[[0.01785488 0.01877308 0.00871841 0.94005877 0.00505726 0.00584777
  0.0036898 ]]
1/1 [==============================] - 0s 373ms/step
Dermatofibroma
```

Рисунок 12. Программная реализация функции прогноза и распределения вероятностей

В ходе разработки был создан словарь с закодированными целевыми классами на этапе предварительной обработки и трансформации датасета метаданных. Закодированные классы представлены в словаре, с помощью метода модели predict мы предсказали вероятности принадлежности фотоснимка к тому или иному классу, для первого мы видим, что наибольшая вероятность равна 0,84 с индексом 4, для второго 0,94 с индексом 3, с помощью метода argmax мы получили индекс максимального значения вероятности в массиве, соотвественно равны 4 и 3, по полученному индексу и равному ключу словаря опредлили описание диагноза, для первого модель точно предсказала меланоцитарный невус, для второй дерматофиброму с вероятностями 84% и 94% соотвественно. Для конечного пользователя важно интерпретировать результаты нейросети, как она пришла к этому исходу, поскольку зачастую она является моделью "черного ящика", поэтому гистограмма по распределению вероятностей покажет почему модель пришла к этому результату, однако более информативным для интерпретации результатов нейросетей является построение тепловых карт GradCam, показывающих на каких ключевых областях внимания изображения были сделаны выводы. Для реализации гистограммы мы сделали прогноз и получили список индексов и значений вероятностей, создали словарь, где ключ это индекс, а значение это вероятность для каждого класса, для удобства создали датафрем с ключом и значением вероятности. С помощью метода датафрема map мы по индексу, равному ключу описательного словаря, получили наименование классов. С помощью библиоткеи matplotlib.pyplt мы построили гистограммы распределения вероятностей рис. 13.

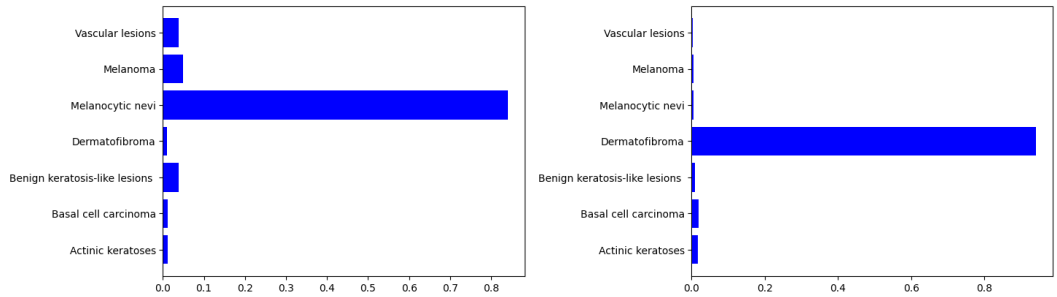


Рисунок 13. Гистограммы распределения вероятностей

На гистограммах видно, какой класс является наиболее вероятным. Полученные результаты исследования и модели будут использованы для разработки

интеллектуальной адаптивной рекомендательной системы, способной перенастраиваться и переобучаться для детектирования новых случаев. На рисунке 14 показан интерфейс окна диагностики по фотоснимкам.

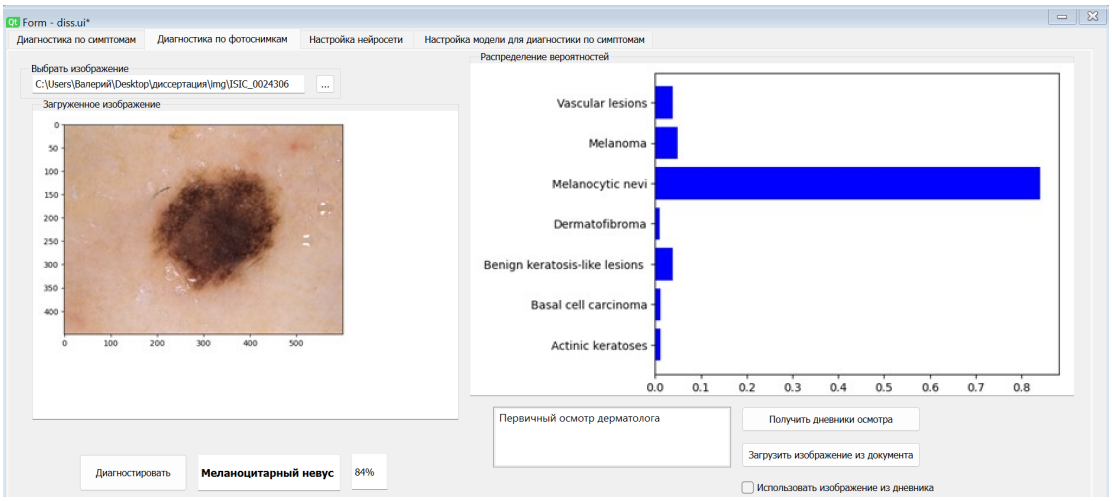


Рисунок 14. Интерфейс окна диагностики по фотоснимкам

Таким образом, мы выработали алгоритм к формированию эффективной архитектуры СНС на основе ансамбля с средневзвешенным объединением результатов, выработаны рекомендацции к формированию эффективной базовой архитектуры СНС, проведено исследование ансамбля с применением предварительно обученных нейросетей на ImageNet с глубокой архитектурой на основе простого усреднения и с применением метода трансферного обучения на малом объеме обучающей выборки, в результате модель переобучилась и показала низкий результат по сравнению с дургими моделями, эту проблему удалось решить обогащением выборки и равномерным распределением обучающей и тестовой выборки по каждому классу.

Заключение

В ходе исследования мы реализовали несколько ансамблей СНС, с применением алгоритмов усреднения и объединения результатов моделей, первый алгоритм на основе простого арифметического усреднения моделей ансамбля оказался не оптимальным в применении к решаемой задачи, поскольку использует НС с очень глубокой архитектурой, обучение производилось на небольшой выборке, поэтому произошло переобучение сети, вследствие чего понизилось качество итогового классификатора, точность достигла 74% на тестовой выборке. Вторая модель реализована на основе ансамбля моделей с эффективной архитектурой НС и с средневзвешенным усреднением результата, она показала результат лучше, чем предыдущая модель и достигла точности в 83%, что на 9% выше. Таким образом, мы экспериментальным путем апробировали результаты построенных моделей, выработали алгоритм к формирования эффективного ансамбля СНС с применением методов усреднения, исследовали влияние переобучения сети на конечный результат, сформировали рекомендации к построению эффективного базового классификатора СНС и построили на основе алгоритмов архитектуры и модели СНС, полученные результаты будут использованы для разработки интеллектуальной системы поддержки принятия врачебных решений.

Библиография

1. Thoma M. Analysis and optimization of convolutional neural network architectures, 2017.

2. Cruz Y. J. et al. Ensemble of convolutional neural networks based on an evolutionary algorithm applied to an industrial welding process //Computers in Industry. – 2021. – Т. 133. – С. 103-530.
3. Yang S. et al. An ensemble classification algorithm for convolutional neural network based on AdaBoost //2017 IEEE/ACIS 16th International Conference on Computer and Information Science (ICIS). – IEEE, 2017. – С. 401-406.
4. Basili V. R., Briand L. C., Melo W. L. A validation of object-oriented design metrics as quality indicators //IEEE Transactions on software engineering. – 1996. – Т. 22. – №. 10. – С. 751-761.
5. Нейронные сети. Переобучение-что это и как этого избежать, критерии останова обучения. [Электронный ресурс]. URL: https://proproprogs.ru/neural_network/pereobuchenie-chto-eto-i-kak-etogo-izbezhat-kriterii-ostanova-obucheniya (дата обращения 09.02.2024).
6. Воронецкий Ю. О., Жданов Н. А. Методы борьбы с переобучением искусственных нейронных сетей // Научный аспект. 2019. №2. [Электронный ресурс] URL: https://na-journal.ru/2-2019-tehnicheskie-nauki/1703-metody-borby-s-pereobucheniem-iskusstvennyh-neironnyh-setei (дата обращения: 10.02.2024).
7. Li C. et al. Improving forecasting accuracy of daily enterprise electricity consumption using a random forest based on ensemble empirical mode decomposition // Energy. – 2018. – Т. 165. – С. 1220-1227.
8. Omisore O. M. et al. Weighting-based deep ensemble learning for recognition of interventionalists' hand motions during robot-assisted intravascular catheterization // IEEE Transactions on Human-Machine Systems. – 2022. – Т. 53. – №. 1. – С. 215-227.
9. Ансамблирование моделей нейронных сетей с использованием библиотеки Keras. [Электронный ресурс]. URL:https://se.moevm.info/lib/exe/fetch.php/courses:artificial_ neural_ networks:pr_8.pdf (дата обращения 11.02.2024).
10. Метод оптимизации Нелдера — Мида. Пример реализации на Python. [Электронный ресурс]. URL:https://habr.com/ru/articles/332092/ (дата обращения 09.02.2024).
11. Клюева И. А. Методы и алгоритмы ансамблирования и поиска значений параметров классификаторов. [Электронный ресурс]. URL:https://dissov.pnzgu.ru/files/dissov.pnzgu.ru/2021/tech/klyueva/ dissertaciya_ klyuevoy _i_a_.pdf (дата обращения 08.02.2024).
12. Микрюков, А. А. Классификация событий в системах обеспечения информационной безопасности на основе нейросетевых технологий / А. А. Микрюков, А. В. Бабаш, В. А. Сизов // Открытое образование. – 2019. – Т. 23. № 1. – С. 57-63.
13. Gizluk D. Adaptive optimization methods. // Neural networks are simple (part 7). 2020. №7. [Электронный ресурс]. URL:https://www.mql5.com/ru/articles/8598#para21 (дата обращения: 10.02.2024).
14. Mason L. et al. Boosting algorithms as gradient descent //Advances in neural information processing systems. – 1999. – Т. 12.
15. Zaheer R., Shaziya H. A study of the optimization algorithms in deep learning //2019 third international conference on inventive systems and control (ICISC). – IEEE, 2019. – С. 536-539.
16. Староверов Б. А., Хамитов Р. Н. Реализация глубокого обучения для прогнозирования при помощи ансамбля нейронных сетей //Известия Тульского государственного университета. Технические науки. – 2023. – №. 4. – С. 185-189.
17. Onan A., Korukoğlu S., Bulut H. A multiobjective weighted voting ensemble classifier

based on differential evolution algorithm for text sentiment classification // Expert Systems with Applications. – 2016. – Т. 62. – С. 1-16.

18. Kim H. et al. A weight-adjusted voting algorithm for ensembles of classifiers //Journal of the Korean Statistical Society. – 2011. – Т. 40. – №. 4. – С. 437-449.
19. Yao X., Islam M. M. Evolving artificial neural network ensembles //IEEE Computational Intelligence Magazine. – 2008. – Т. 3. – №. 1. – С. 31-42.
20. Anand V. et al. Weighted Average Ensemble Deep Learning Model for Stratification of Brain Tumor in MRI Images //Diagnostics. – 2023. – Т. 13. – №. 7. – С. 1320.
21. The International Skin Imaging Collaboration. [Электронный ресурс].-URL: https://www.isic-archive.com (дата обращения 12.02.2024).
22. Alexandropoulos S. A. N., Kotsiantis S. B., Vrahatis M. N. Data preprocessing in predictive data mining // The Knowledge Engineering Review. – 2019. – Т. 34. – С. e1.
23. García S., Luengo J., Herrera F. Data preprocessing in data mining. – Cham, Switzerland: Springer International Publishing, 2015. – Т. 72. – С. 59-139.
24. Liang G., Zheng L. A transfer learning method with deep residual network for pediatric pneumonia diagnosis // Computer methods and programs in biomedicine. – 2020. – Т. 187. – С. 104-964.
25. InceptionV3. [Электронный ресурс].-URL: https://keras.io/api/ applications/inceptionv3/ (дата обращения 13.02.2024).
26. InceptionResnNetV2. [Электронный ресурс]. URL: https://keras.io/api/ applications/inceptionresnetv2/ (дата обращения 13.02.2024).
27. VGG16. [Электронный ресурс]. URL: https://keras.io/api/ applications/vgg/#vgg16-function (дата обращения 13.02.2024).
28. Щетинин Е. Ю. О некоторых методах сегментации изображений с применением свёрточных нейронных сетей // Информационно-телекоммуникационные технологии и математическое моделирование высокотехнологичных систем. – 2021. – С. 507-510.
29. Rosebrock A. Change input shape dimensions for fine-tuning with Keras. // AI & Computer Vision Programming. 2019. [Электронный ресурс]. URL:https://pyimagesearch.com/2019/06/24/ change-input-shape-dimensions-for-fine-tuning-with-keras/ (дата обращения 14.02.2024).
30. Костин К. А. и др. Адаптивный классификатор патологий для компьютерной диагностики заболеваний с использованием сверточных нейронных сетей по медицинским изображениям и видеоданным: магистерская диссертация по направлению подготовки: 01.04. 02-Прикладная математика и информатика. – 2017.
31. A. Krizhevsky, I. Sutskever, G.E. Hinton. ImageNet Classification with Deep Convolutional Neural Networks. Proceedings of Advances in Neural Information Processing Systems 25 (NIPS 2012), 2012, Pp. 1097-1105.
32. Wang J., Lin J., Wang Z. Efficient hardware architectures for deep convolutional neural network // IEEE Transactions on Circuits and Systems I: Regular Papers. – 2017. – Т. 65. – №. 6. – pp. 1941-1953.
33. Phung V. H., Rhee E. J. A high-accuracy model average ensemble of convolutional neural networks for classification of cloud image patches on small datasets //Applied Sciences. – 2019. – Т. 9. – №. 21. – С. 4500.
34. The differential evolution method. [Электронный ресурс]. URL: https://docs.scipy.org/ doc/scipy/reference/generated/scipy.optimize.differential_evolution.html (дата обращения: 13.02.2024).

35. Как разработать средневзвешенный ансамбль для глубоких обучающих нейронных сетей. // 2018. [Электронный ресурс]. URL: https://machinelearningmastery.ru/weighted-average-ensemble-for-deep-learning-neural-networks/# (дата обращения: 13.02.2024)

Результаты процедуры рецензирования статьи

В связи с политикой двойного слепого рецензирования личность рецензента не раскрывается.
Со списком рецензентов издательства можно ознакомиться здесь.

Рецензируемая работа посвящена исследованию и разработке алгоритмов формирования эффективного ансамбля сверточных нейронных сетей для классификации изображений путем решения задача многоклассовой одновариантной классификации изображений на основе обучения модели с учителем на размеченном наборе данных.
Методология исследования базируется на применении кибернетического подхода и концепции искусственного интеллекта к распознаванию образов с использованием искусственных нейронных сетей, созданием ансамблей моделей машинного обучения. Эксперименты по классификации изображений были проведены на материалах открытого источника данных по дерматологическим заболеваниям, представленных в формате csv, исследование проведено на языке python с аппаратным графическим ускорителем и использованием фреймворков машинного обучения и разработки нейронных сетей (keras, tensorflow, sklearn).
Актуальность работы авторы справедливо связывают с тем, что для повышения точности прогноза и уменьшения ошибок можно использовать ансамбли моделей нейронных сетей, которые находят широкое применение в разных сферах, включая распознавание изображений в интеллектуальных системах поддержки принятия врачебных решений.
Научная новизна рецензируемого исследования, по мнению рецензента состоит в разработке алгоритмов ансамблевых моделей сверточных нейронных сетей с применением методов усреднения и рекомендациях по применению полученных результатов для разработки интеллектуальной системы поддержки принятия врачебных решений.
В тексте статьи выделены следующие разделы: Введение, Материалы и методы, Результаты и обсуждение, Заключение, Библиография.
Во введении обоснована актуальность темы, отмечены преимущества и недостатки ансамблирования в машинном обучении. Далее обозначена проблема переобучения в использовании глубоких нейронных сетей на малом объеме данных – явление, когда сеть очень хорошо прогнозирует на обучающей выборке и плохо на тестовой, в результате чего наблюдается большая разница показателя качества на обучающей и валидационной выборке. В публикации рассмотрен принцип работы ансамбля, заключающийся в том, что каждая нейронная сеть обрабатывает входные данные независимо и выдает свой собственный прогноз, а затем, путем агрегации результатов всех нейронных сетей, получается конечный прогноз, в котором индивидуальные ошибки компенсируются, а общая производительность системы улучшается. Приведены несколько способов объединения результата в ансамбле: простое голосование, взвешенное голосование, смешанное голосование; представлено сравнение полученных оценок точности трех моделей. Статья иллюстрирована двумя таблицами, 14 рисунками, содержит 6 формул.
Библиографический список включает 35 источников – научные публикации отечественных и зарубежных авторов на русском и английском языках, а также интернет-ресурсы по рассматриваемой теме, на которые в тексте приведены адресные

ссылки, подтверждающие наличие апелляции к оппонентам.

Из резервов улучшения публикации можно отметить следующие. Во-первых, авторам предлагается рассмотреть вариант корректировки названия статьи без предлога «к»: «Исследование и разработка алгоритмов формирования…». Во-вторых, заголовки таблиц лучше разместить в соответствии с принятыми правилами – перед таблицами, а не после них.

Рецензируемый материал соответствует направлению журнала «Программные системы и вычислительные методы», отражает ход и результаты проведенной авторами работы по созданию системы искусственного интеллекта, вызовет интерес у читателей, а поэтому после некоторой доработки в соответствии с высказанными пожеланиями статья рекомендуется к опубликованию.

Программные системы и вычислительные методы
Правильная ссылка на статью:
Черепенин В.А., Кацупеев А.А. Анализ подходов к созданию системы «Умная теплица» на основе нейронной сети // Программные системы и вычислительные методы. 2024. № 1. DOI: 10.7256/2454-0714.2024.1.69794 EDN: XAZVOW URL: https://nbpublish.com/library_read_article.php?id=69794

Анализ подходов к созданию системы «Умная теплица» на основе нейронной сети

Черепенин Валентин Анатольевич

ORCID: 0000-0002-6310-1939

аспирант, кафедра Информационные и измерительные системы и технологии, Южно-Российский государственный политехнический университет (НПИ) имени М.И. Платова

346428, Россия, Ростовская область, г. Новочеркасск, ул. Просвещения, 132

cherept2@gmail.com

Кацупеев Андрей Александрович

кандидат технических наук

доцент, кафедра Информационные и измерительные системы и технологии, Южно-Российский государственный политехнический университет (НПИ) имени М.И. Платова

346428, Россия, Ростовская область, г. Новочеркасск, ул. Просвещения, 132

andreykatsupeev@gmail.com

Статья из рубрики "Системный анализ, поиск, анализ и фильтрация информации"

DOI:

10.7256/2454-0714.2024.1.69794

EDN:

XAZVOW

Дата направления статьи в редакцию:

08-02-2024

Аннотация: Исследование затрагивает важную тему разработки и внедрения интеллектуальных систем в агропромышленном производстве, фокусируясь на создании "Умной теплицы" с применением нейронных сетей. В работе детально анализируются ключевые технологические инновации и их роль в устойчивом сельском хозяйстве. Особое внимание уделяется изучению методов сбора, обработки и анализа данных для оптимизации условий выращивания растений. Рассматриваются вопросы эффективности

использования ресурсов, контроля влажности, температуры, уровня углекислого газа и освещённости, а также автоматизация полива и подачи удобрений. Особое внимание уделяется созданию адаптивных алгоритмов для прогнозирования оптимальных условий, повышающих урожайность и качество сельскохозяйственной продукции, при одновременном снижении экологического воздействия и затрат. Это открывает новые перспективы для устойчивого развития аграрного сектора, способствуя более эффективному и экологически чистому сельскому хозяйству. Исследование реализовано с использованием аналитического обзора литературы, сравнительного анализа существующих решений и моделирования работы нейронных сетей для предсказания оптимальных условий выращивания. Исследование представляет собой значительный вклад в область применения искусственного интеллекта для управления микроклиматом в теплицах, демонстрируя возможности нейронных сетей в автоматизации агропромышленных процессов. Анализируются перспективы использования ИИ для предсказания и оптимизации условий выращивания, что может привести к революционным изменениям в сельском хозяйстве. Выявленные научные новизны включают разработку и тестирование алгоритмов прогнозирования, способных адаптироваться к меняющимся внешним условиям, и обеспечивать максимальную продуктивность при минимальных затратах ресурсов. Выводы исследования подчеркивают важность дальнейшего изучения и внедрения интеллектуальных систем в агропромышленности, указывая на их потенциал в увеличении урожайности и улучшении качества продукции при одновременном снижении экологического воздействия. В заключении, авторы оценивает перспективы применения нейронных сетей в агропромышленном секторе и рассматривает возможные пути дальнейшего развития "Умных теплиц".

Ключевые слова:

интернет вещей, теплица, биотехнология, глубокое обучение, гибридная нейронная сеть, микроклимат, алгоритм оптимизации, нейронные сети, Умная теплица, Анализ подходов

Введение

С развитием технологий интернета вещей (далее *IoT*) и искусственного интеллекта (ИИ) системы «Умная теплица» становятся важным направлением в современном сельском хозяйстве. Эффективное использование ресурсов, оптимальный контроль климата и автоматизация процессов выращивания растений предоставляют сельскохозяйственным предприятиям новые возможности для повышения урожайности и снижения затрат. В данной статье проводится обзор существующих подходов к созданию системы «Умная теплица», с акцентом на использование нейронных сетей. Исследование направлено на выявление ключевых аспектов и преимуществ, которые предоставляют нейронные сети в контексте управления тепличным климатом. Рассматриваются методы сбора и анализа данных, внедрение сенсорных технологий, а также алгоритмы обработки информации с применением нейронных сетей для оптимизации условий роста растений. Этот обзор предоставляет комплексное понимание современного состояния технологий «Умной теплицы» и выделяет перспективы дальнейших исследований в данной области.

Исследование и сравнение методов климат-контроля в теплицах

Климат в теплице основан на условиях окружающей среды, необходимых растениям для

жизни. Микроклимат теплицы очень сложный, нелинейный, многопараметрический, и зависит от набора внутренних и внешних факторов, включая метеорологические факторы, такие как влажность и температура окружающей среды, интенсивность солнечного излучения, скорость и направление ветра.

Составляющие теплицы — это внутренние факторы, различные сельскохозяйственные культуры, компоненты теплицы и такие элементы, как запотевание, отопление, вентиляционные системы, типы почвы и т.д. Различное описание тепличного климата объясняется в двух подходах: один основан на уравнениях массового расхода энергии, которые описывают процесс. Другой использует подход системной идентификации, состоящий из исследования как входных, так и выходных данных процесс [1]. Таким образом, данные подходы созданы для решения всех проблем микроклимата на основе современных методов адаптивных и нелинейных систем. Для достижения заранее определенных и оптимальных результатов климат-контроль в теплицах создает благоприятную среду для выращивания. Для управления окружающей средой теплицы было предложено несколько стратегий и методов управления, таких как адаптивное управление, прогнозирующее управление, нечеткая логика управления, надежное управление, нелинейное управление с обратной связью и оптимальное управление. Было проведено множество исследований, в которых ведется поиск экономии энергии, где применяются методы оптимизации сбора энергии, физические модели и методы вычислительной гидродинамики для прогнозирования микроклимата теплиц [2, 3].

Эти методы прогнозирования делятся на две группы:

1. Физический метод.

2. Метод черного ящика.

Первый метод основан на математической теории, которая необходима для регулирования огромного количества параметров, и расчет этих параметров затруднен. Метод черного ящика основан на современных вычислительных технологиях, которые не всегда обеспечивают сходимость к оптимальному решению и даже легко проходят частичную оптимизацию.

С другой стороны, нейронные сети, в отличие от запрограммированных, учатся реализовывать все закономерности. Эти тепличные системы очень уместны для отражения знаний, которые нельзя запрограммировать для представления нелинейных явлений [4].

Обучение и применение нейронных сетей в сельском хозяйстве

Поскольку теплицы неизменны во времени, нелинейны и имеют сильную связь, проводится множество исследований, чтобы выбрать искусственные нейронные сети (далее *ANN*) для моделирования, оптимизации, прогнозирования и управления всеми этими процессами. В этом разделе рассматриваются все различные приложения и исследования *ANN* в области тепличных технологий. Разработка модели этого типа, в будущем расширит возможности ее применения и объединится с технологиями, которые востребованы в интеллектуальном сельском хозяйстве. На рис. 1 показывается классификация моделей различных теплиц.

Рис. 1. – Классификация моделей теплицы

ANN — это алгоритм машинного обучения, основанный на анализе и понимании человеческого нейрона. Это вычислительная модель с биологическим влиянием, состоящая из обрабатывающих элементов (нейронов) и взаимосвязей между ними с коэффициентами, обычно называемыми весами [5].

Нейроны получают входные данные в виде импульса. Некоторыми мерами, используемыми для описания активности нейронов, являются пиковая скорость, создаваемая с течением времени, и средняя пиковая генерация в нескольких прогонах. Нейрон распознается по скорости, с которой он производит пики. С помощью адаптивных синаптических весов нейроны связаны с другими нейронами предыдущего слоя. В наборе весов соединения обычно хранятся знания. Процесс обучения выявляет подходящий метод обучения, когда эти веса соединений изменяются соответствующим образом. Эти методы обучения содержат входные данные для сети и генерируют желаемый результат путем изменения весов для получения идеального результата. По мере обучения методы веса будут иметь соответствующую информацию по сравнению с предыдущим обучением, поскольку он содержит всю избыточную и бессмысленную информацию [6].

В нейронных сетях обучение является важной частью. Обучение модели нейронной сети — это процесс настройки весов и смещений сети таким образом, чтобы она могла точно предсказать выходные данные для заданных входных данных. Это делается путем многократного представления сети примеров входных и выходных данных, а затем корректировки весов и смещений таким образом, чтобы прогнозы сети становились более точными. Процесс обучения повторяется много раз до тех пор, пока модель не сойдется, что означает, что ошибка больше не уменьшается значительно. Количество повторений процесса обучения называется количеством эпох. Этот процесс определит связь ввода и вывода, поскольку для этого требуется наиболее точный прогнозирующий расчет. Этот процесс обучения подразделяется на две категории:

1. Под наблюдением;

2. Без присмотра.

Обучение под наблюдением воспринимает ожидаемые результаты и маркирует данные. При неконтролируемом обучении не обязательно воспринимать данные, так как обучение осуществляется посредством определения местоположения представления данных и

внутренних структур.

Структура сети зависит от типа описываемой задачи, сложности системы и процесса обучения. В таблице №1 показаны различные виды нейронных моделей, подробно описаны входные и выходные переменные, которые использовались, архитектура сети, функции активации, которые использовались сетью или каждым уровнем сети, и алгоритм, используемый в тренировочный процесс [7].

Таблица №1. Типы нейросетевых моделей для прогнозирования

№ Модели	Входные переменные	Выходные переменные	Тип	Функции активации	Метод обучения	% оц
1	Наружная температура; Наружная влажность; Скорость ветра; Солнечная радиация;	Внутренняя температура; Внутренняя влажность	*FFNN*	Передаточная функция Гаусса для скрытого слоя	Обратное распространение градиентного спуска (далее *BP*)	5.(
2	Температура наружного воздуха; Наружная влажность; Скорость ветра; Солнечная радиация; Температура воздуха внутри	Внутренняя влажность	*FFNN*	Сигмовидная передаточная функция для скрытого слоя	*BP*	15
3	Температура наружного воздуха; Солнечное излучение; Влажность внутри помещения	Внутренняя температура	*FFNN*	Сигмовидная передаточная функция для скрытого слоя	Ортогональный метод наименьших квадратов	20
4	Внешняя температура; Внешняя гигрометрия; Глобальный радиант;	Внутренняя температура; Внутренняя гигрометрия	*RNN*	Сигмовидная функция для скрытого слоя	Метод обучения с учителем	20

	Скорость ветра					
5	Внешняя температура; Внешняя гигрометрия; Обогрев; Сдвижной затвор в градусах; Опрыскиватель	Внутренняя температура; Внутренняя гигрометрия	*RNN*	Сигмовидная функция для скрытого и выходного слоев	Глубокое обучение, где использовался алгоритм *BP*	30
6	Внешняя температура; Внешняя влажность; Внутренняя температура; Внутренняя влажность	Внутренняя температура; Внутренняя влажность	*NNARX*	Функция гиперболического тангенса для скрытого слоя	Левенберг-Марквардт	25

Обучение нейронной сети сводится к минимизации функции ошибки, путем корректировки весовых коэффициентов синоптических связей между нейронами. Под функцией ошибки понимается разность между полученным ответом и желаемым. Для нейронов выходного слоя известны их фактические и желаемые значения выходов. Поэтому настройка весов связей для таких нейронов является относительно простой [8]. Однако для нейронов предыдущих слоев настройка не столь очевидна. Долгое время не было известно алгоритма распространения ошибки по скрытым слоям. Величина ошибки определяется по следующей формуле:

$$E_p = \frac{1}{2}\sum_j \left(t_{pj} - y_{pj}\right)^2,$$

где *Ep* – величина функции ошибки для образа *p*; *tpj* – желаемый выход нейрона *j* для образа *p*; *ypj* – активированный выход нейрона *j* для образа *p*.

Метод обучения с учителем включает в себя сокращение функции стоимости, которая собирает ошибки между желаемыми выходными данными и фактическими выходными данными для заданных входных данных. Чтобы уменьшить эту функцию стоимости, используется несколько методов. Алгоритм *BP* является одним из наиболее часто используемых для получения результатов в однослойных и многослойных сетях. Алгоритм предназначен для самоорганизации *ANN* и получен по закону Хебба. Выявленные методы обучения позволили разработать продвинутые алгоритмы, такие как самоорганизующиеся карты (далее *S.O.M.*) и самоорганизующийся древовидный алгоритм (далее *SOTA*). Основанные на неконтролируемых данных, это алгоритмы кластеризации временных рядов. Для визуализации и кластеризации данных используется инструмент *S.O.M.* Основным недостатком этого инструмента является то, что пользователю необходимо выбирать размер карты, что приводит к многочисленным экспериментам с картами разного размера в попытке получить оптимальный результат.

Этот процесс может быть довольно медленным. Более точная классификация паттернов в последнем слое *SOTA* позволит классифицировать на начальных этапах группу паттернов, отделенных друг от друга. Используя методы *SOTA*, нейронные сети могут быть построены автоматически из доступных обучающих данных.

Согласно этим методам применения и обработки инструментов, переменные, связанные с климатом для теплицы, имеют решающее значение. Некоторые параметры, такие как расчет скорости, прогнозирование поведения точности и управление переменными различными элементами, остаются огромной проблемой. Используя некоторые нелинейные методы, искусственные нейронные сети в значительной степени решают эти задачи.

Оптимизация микроклимата в теплицах с использованием нейронных сетей

Для достижения контролируемого сельскохозяйственного производства, теплицы представляют собой нелинейные и сложные системы. Эти системы имеют внутренние факторы, составной динамический импульс внешних факторов и механизмы управления. Основанный на микроклимате и управлении тепличными культурами, он выступает центральным элементом и важнейшей частью системы. Эта система, как правило, учитывает некоторые общие вопросы, которые являются более важными реакциями микроклимата, поскольку он представляет собой комплекс и разнообразие культур в теплице. Тепличная система в основном фокусируется на наборе окружающей среды, которая влияет на развитие и рост сельскохозяйственных культур. Система микроклимата в теплицах в последние годы привлекла к себе пристальное внимание из-за значительного вклада в улучшение урожайности. Для прогнозирования различных элементов, таких как количество *CO*2, температуры и относительной влажности воздуха, в систему внедрили статистику, инженерию и искусственный интеллект.

В тепличной системе используется обычный контроль, чтобы гарантировать желаемую производительность, но этот контроль может быть неудовлетворительным. В последнее время интерес вызывает тема применения нейронных сетей для управления микроклиматом. Чтобы отразить нелинейные характеристики, нейросеть предоставляет все надежные модели для теплицы, поскольку с помощью традиционных методов трудно решить, поскольку надежные модели не требуют каких-либо предварительных знаний о системе, в режиме реального времени модель вполне удовлетворительно работает для динамических систем [9].

Наиболее важными атрибутами в системе микроклимата теплицы являются влажность и температура. В окружающей среде может происходить сложный газообмен, а также взаимодействие между массой, теплом и внутренним воздухом, а также некоторыми элементами теплицы с внешней стороны. Построение надежной модели — очень сложная задача, поскольку она должна выполнять все параметры, математические функции и функции преобразования [10]. Когда мы строим модель с помощью *ANN*, видим большие возможности для отображения всех нелинейных функций для создания многочисленных систем производственных процессов. Чтобы построить модель, нужно спроектировать сеть системы, такую как влажность и температура воздуха в теплицах, которые рассматриваются как выходные данные; это вызвано некоторыми фактами. Для модели самое сложное — установить входные данные, чтобы лучше понять систему. Для рассмотрения входной переменной в системе необходимо следовать следующим соображениям:

1. Характер физической зависимости входа и выхода.

2. Корреляция между входами и выходами.

3. Диапазон переменных входа.

Входными переменными системы являются внутренняя относительная влажность воздуха, внутренняя температура, внутреннее атмосферное давление, внешняя температура, внешнее атмосферное давление, внешняя температура, скорость и направление ветра. Выходной переменной системы является *CO*2. Во всех слоях используется передаточная функция. 0,97 коэффициента получено с использованием прогноза концентрации *CO*2. Искусственная нейронная сеть должна быть обучена в соответствии со всеми возможными ситуациями и измеримыми объектами, чтобы получить уникальный результат с ограниченными данными [11].

Для правильного функционирования нейронной сети требуется база данных. В данном разделе описано множество моделей, в которых эти искусственные нейронные сети используются для прогнозирования микроклимата тепличной системы, а оптимизация сети связана с ее структурой и типом. Поскольку для модели требуются обучающие данные, сбор данных является одним из ключевых моментов, который имеет одинаковое значение для системы для эффективного вывода. В каждой базе данных будет три общих этапа, таких как тестирование, обучение и проверка. Эти фазы необходимо учитывать в каждой нейронной сети. Модель Бэкбокса требует огромного количества данных в *ANN* [12, 13]. Необязательная сумма в модели черного ящика — это метод проб и ошибок, который выбирает скрытый слой и нейроны. Поскольку рассматриваем огромный объем данных для получения точного вывода, это приводит к снижению эффективности процесса. Некоторые методы применяются к модели с огромными данными, которые можно упростить и оптимизировать процесс обучения *N.N.* База данных за 12 месяцев является идеальной для *ANN* для прогнозирования нейронной сети теплицы, которая имеет огромное количество данных. Была выбрана модель в течение 29 дней: 22 дня фазы обучения и 7 дней фазы тестирования. Модель считалась фазами, разделенными на 15% для проверки, 15% для тестирования и 70% для обучения сети. На основе модели и требований считается, что период дает точный результат. Рис. 2 показывает прогноз микроклимата теплицы в искусственной нейронной сети.

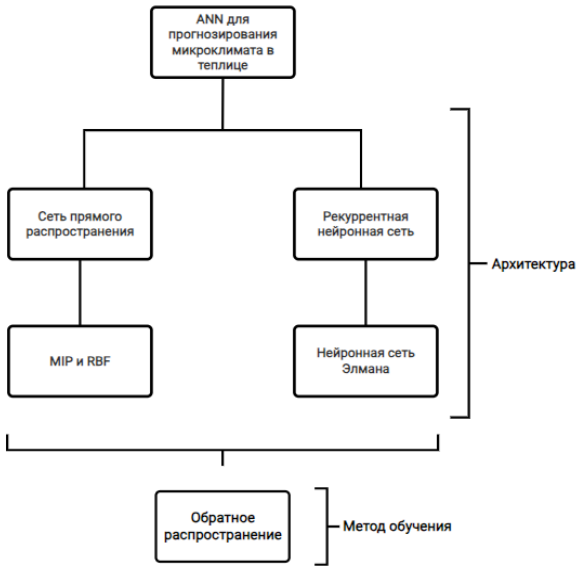


Рис. 2. - Прогноз микроклимата теплицы в искусственной нейронной сети

Заключение

В отличие от физических моделей, *ANN* требуется всего несколько минут, чтобы завершить прогноз климата в помещении, учитывая, что задействовано много неизвестных факторов, которые невозможно изучить с помощью физических моделей. Комбинация *ANN* и физических моделей позволила бы лучше прогнозировать микроклимат, однако конструкция этой гибридной сети мало изучена и, следовательно, эта сеть должна быть изучена. В рамках других приложений нейронных сетей в теплицах оценка и прогнозирование эпидемий и болезней сельскохозяйственных культур может осуществляться с помощью таких технологий, как анализ изображений в сочетании с моделями глубокого обучения, такими как *CNN*. Точно так же с помощью этих методов может быть осуществимо прогнозирование микроклимата, поскольку эти методы могут обрабатывать больший объем информации по сравнению с традиционными моделями *ANN*. Роль *A N N* заключается в разработке прогностических моделей, использующих преимущества генерируемой информации и управления ею.

Библиография

1. Taki, M. "Solar thermal modeling and application in greenhouses"/ M. Taki, A. Rohani, M. Rahmati-Joneidabad // Info Proc Agri, 5 (2018), pp. 83-113. DOI:10.1016/J.inpa.2017.10.003.
2. Huihui, Yu "Temperature prediction in a Chinese solar greenhouse based on LS-SVM optimized with improved PSO" / Huihui Yu, Yinyi Chen, Shahbaz Gul Hassan, Daoliang Li// Comput Electron Agric, 122 (2016), pp. 94-102. DOI:10.1016/J.compag.2016.01.019.
3. Taki, M. "The relationship of energy and yield costs and sensitivity analysis for growing tomatoes in greenhouses in Iran" / M.Taki, R.Abdi, M.Akbarpour, H.Ghasemi-Mobtaker // Agric Eng Int: CIGR J, 15 (2013), pp. 59-67. ISSN:16821130.
4. Abdel Ghani, A.M. "The use of solar energy by greenhouses: general relations" / A.M. Abdel Ghani, I.M. Helal // Renewable Energy, 36 (2011), pp. 189-196. DOI:10.1016/j.renene.2010.06.020.
5. Nielsen, H. "Identification of transfer functions for regulating air temperature in a greenhouse" / H.Nielsen, P.Madsen // J Agric Eng Res, 60 (1995), pp. 25-34. DOI:10.1006/jaer.1995.1093.
6. Dariushi, E. "Labyrinth prediction of internal parameters of a tomato greenhouse in a semi-arid zone using a time series model of artificial neural networks" / E. Dariushi, K. Aassif, L. Lekush, G. Buirden // Measurement, 42 (2009), pp. 456-463. DOI:10.1016/J.measurement.2008.08.013
7. Кацупеев, А. А. Постановка и формализация задачи формирования информационной защиты распределённых систем / А. А. Кацупеев, Е. А. Щербакова, С. П. Воробьев // Инженерный вестник Дона. – 2015, 34, с. 21. – EDNTXTMHJ.
8. Abdi, R. "Analysis of energy consumption and greenhouse gas emissions from agricultural production" / R. Abdi, M. Taki, M. Akbarpour // Int J Nat Eng Sci,6 (2012), pp. 73-79. ISSN: 026322410.
9. Ruano, A.E. "Forecasting building temperature using neural network models" / A.E. Ruano, E.M. Crispim, E.Z.E. Conceicao, M.Lucio // Energy Build, 38 (2006), pp. 682-694. DOI:10.1016/J.enbuild.
10. He, F. "Modeling of air humidity in a greenhouse using an artificial neural network and analysis of the main components" / F. He, S. Ma // Prog Electron-X. Sciences, 71 (2010), pp. 19-23. DOI: 10.1016/J.compag.2009.07 0.011.
11. Taki, M. "Models of heat transfer and MLP neural networks for predicting internal

environmental variables and energy losses in a semi – solar greenhouse" / M. Taki, Yu. Ajabshirchi, S. F. Ranjbar, A. Rohani, M. Matlub // Energy Build, 110 (2016), pp. 314-329. DOI:10.1016/j.enbuild.2015.11.010.

12. Ilyas, S.A. "Neural network logic sensor RBF for monitoring technological emissions" / Ilyas S.A., Elshafey M., Habib M.A. // Control Eng Practice, 21 (2013), pp. 962-970. DOI:10.1016/J.conengprac.2013.01.007.
13. Воробьев, С. П. Исследование модели транзакционной системы с репликацией фрагментов базы данных, построенной по принципам облачной среды / С. П. Воробьев, В. В. Горобец // Инженерный вестник Дона, 2012, 22, с. 49. – EDNPRXKMH

Результаты процедуры рецензирования статьи

В связи с политикой двойного слепого рецензирования личность рецензента не раскрывается.
Со списком рецензентов издательства можно ознакомиться здесь.

Статья посвящена анализу разработки систем "Умная теплица" с применением нейронных сетей в контексте современного сельского хозяйства. Авторы сосредотачиваются на важности интеграции технологий IoT и ИИ для повышения эффективности агропредприятий через оптимизацию ресурсов и автоматизацию процессов. Исследование базируется на анализе существующих моделей и подходов, с акцентом на использование нейронных сетей для управления микроклиматом теплиц и оптимизации условий роста растений. Особое внимание уделено методам обучения нейронных сетей, включая обучение под наблюдением и без присмотра, а также различным архитектурам и функциям активации, используемым в нейросетевых моделях. Актуальность работы подчеркивается растущей потребностью в инновационных технологиях для сельского хозяйства, способствующих повышению урожайности и снижению затрат. Научная новизна заключается в комбинировании нейронных сетей и физических моделей для более точного прогнозирования микроклимата, что позволяет более эффективно управлять агропроцессами. Статья имеет логичную структуру и ясное изложение, начиная с введения и заканчивая выводами и предложениями по дальнейшим исследованиям. В заключении авторы подчеркивают необходимость дальнейшего изучения интеграции ANN в системы управления теплицами и предлагают перспективные направления для будущих исследований, включая применение глубокого обучения для анализа изображений в целях диагностики заболеваний культур. Следует особо отметить упоминание о Модели Бэкбокса, которая представляет собой значимый аспект в контексте исследования. Модель Бэкбокса описывается как метод, требующий значительного объема данных для эффективного обучения искусственных нейронных сетей, используемых в прогнозировании микроклимата теплиц. Этот метод основывается на пробах и ошибках для выбора подходящего скрытого слоя и нейронов, что, как указывается, может привести к снижению эффективности из-за необходимости работы с большими объемами данных. Упоминание Модели Бэкбокса важно для понимания методологических трудностей, с которыми сталкиваются исследователи при работе с нейронными сетями в сельскохозяйственной сфере, а также подчеркивает потребность в оптимизации процесса обучения для повышения эффективности и точности прогнозирования. Это подчеркивает важность выбора и адаптации методов обучения в зависимости от специфики задачи и доступных данных. В целом, статья представляет интерес для широкого круга читателей, включая специалистов в области агрономии, искусственного интеллекта и разработчиков сельскохозяйственных технологий.

Предложенные авторами идеи и методы могут способствовать развитию устойчивого и эффективного сельского хозяйства. В качестве рекомендации для дальнейшего исследования предлагаю разработку и тестирование интегрированных моделей, сочетающих ANN и физические подходы для различных аспектов управления теплицами, а также исследование возможности применения ANN в смежных областях агротехнологий. Это может включать разработку алгоритмов для более точного контроля влажности, температуры, уровня CO2 и других критических параметров, определяющих условия роста растений в теплицах.

Англоязычные метаданные

Integration of cloud, fog, and edge technologies for the optimization of high-load systems

Cherepenin Valentin Anatolyevich

Postgraduate student, Department of Information and Measurement Systems and Technologies, South Russian State Polytechnic University (NPI) named after M.I. Platova

132 Prosveshcheniya str., Novocherkassk, Rostov region, 346428, Russia

cherept2@gmail.com

Smyk Nikolai Olegovich

Postgraduate student, Department of Computer Software, South Russian State Polytechnic University (NPI) named after M.I. Platova

132 Prosveshcheniya str., Novocherkassk, Rostov region, 346428, Russia

smyk.n@list.ru

Vorob'ev Sergei Petrovich

PhD in Technical Science

Associate Professor, Department of Information and Measuring Systems and Technologies, South Russian State Polytechnic University (NPI) named after M.I. Platova

132 Prosveshcheniya str., Novocherkassk, Rostov region, 346428, Russia

vsp1999@yandex.ru

Abstract. The study is dedicated to analyzing methods and tools for optimizing the performance of high-load systems using cloud, fog, and edge technologies. The focus is on understanding the concept of high-load systems, identifying the main reasons for increased load on such systems, and studying the dependency of the load on the system's scalability, number of users, and volume of processed data. The introduction of these technologies implies the creation of a multi-level topological structure that facilitates the efficient operation of distributed corporate systems and computing networks. Modern approaches to load management are considered, the main factors affecting performance are investigated, and an optimization model is proposed that ensures a high level of system efficiency and resilience to peak loads while ensuring continuity and quality of service for end-users. The methodology is based on a comprehensive approach, including the analysis of existing problems and the proposal of innovative solutions for optimization, the application of architectural solutions based on IoT, cloud, fog, and edge computing to improve performance and reduce delays in high-load systems. The scientific novelty of this work lies in the development of a unique multi-level topological structure capable of integrating cloud, fog, and edge computing to optimize high-load systems. This structure allows for improved performance, reduced delays, and effective system scaling while addressing the challenges of managing large data volumes and servicing multiple requests simultaneously. The conclusions of the study highlight the significant potential of IoT technology in improving production processes, demonstrating how the integration of modern technological solutions can contribute to increased productivity, product quality, and risk management.

Keywords: Technology integration, Internet of Things, Scalability, Performance optimization, Edge computing, Fog computing, Cloud computing, High-load systems, Data management,

Service continuity

References (transliterated)

1. Catal, C.; Tekinerdogan, B. Aligning education for the life sciences domain to support digitalization and Industry 4.0. Procedia Computer Science. 2019, 158, 99-106. DOI:10.1016/j.procs.2019.09.032
2. Patel, C.; Doshi, N. A novel MQTT security framework in generic IoT model. Procedia Computer Science. 2020, 171, 1399-1408. DOI:10.1016/j.procs.2020.04.150
3. Subeesh, A.; Mehta, C.R. Automation and digitization of agriculture using artificial intelligence and internet of things. Artificial Intelligence in Agriculture. 2021, 5, 278-291. DOI:10.1016/j.aiia.2021.11.004
4. Faridi, F.; Sarwar, H.; Ahtisham, M.; Kumar, S.; Jamal, K. Cloud computing approaches in health care. Materials Today: Proceedings, 2022, 51, 1217-1223. DOI:10.1016/j.matpr.2021.07.210
5. Tzounis, A.; Katsoulas, N.; Bartzanas, T.; Kittas, C. Internet of Things in agriculture, recent advances and future challenges. Biosystems Engineering. 2017, 164, 31-48. DOI:10.1016/j.biosystemseng.2017.09.007
6. Tao, W.; Zhao, L.; Wang, G.; Liang, R. Review of the internet of things communication technologies in smart agriculture and challenges. Computers and Electronics in Agriculture. 2021, 189, 106352. DOI:10.1016/j.compag.2021.106352
7. Moysiadis, V.; Sarigiannidis, P.; Vitsas, V.; Khelifi, A. Smart farming in Europe. Computer Science Review. 2021, 39, 100345. DOI:10.1016/j.cosrev.2020.100345
8. Raj, M.; Gupta, S.; Chamola, V.; Elhence, A.; Garg, T.; Atiquzzaman, M.; Niyato, D. A survey on the role of Internet of Things for adopting and promoting Agriculture 4.0. Journal of Network and Computer Applications. 2021, 187, 103107. DOI:10.1016/j.jnca.2021.103107
9. Boursianis, A.D.; Papadopoulou, M.S.; Diamantoulakis, P.; Liopa-Tsakalidi, A.; Barouchas, P.; Salahas, G.; Karagiannidis, G.; Wan, S.; Goudos, S.K. Internet of Things (IoT) and agricultural unmanned Aerial Vehicles (UAVs) in smart farming: A comprehensive review. Internet of Things. 2022, 18, 100187. DOI:10.1016/j.iot.2020.100187
10. Singh, S.; Chana, I.; Buyya, R. Agri-Info: Cloud based autonomic system for delivering agriculture as a service. Internet of Things. 2020, 9, 100131. DOI:10.1016/j.iot.2019.10013

Algorithm and software implementation of real-time collaborative editing of graphical schemes using Socket.IO library

Alpatov Aleksey Nikolaevich

PhD in Technical Science

Ph.D. of Engineering Sciences, Docent, Department of Instrumental and Application Software, Federal State Budget Educational Institution of Higher Education «MIREA — Russian Technological University»

78 Vernadsky Avenue, office G-225, Moscow region, 119454, Russia

alpatov@mirea.ru

Iurov Ilia Igorevich

Master student, Department of Instrumental and Application Software, Federal State Budget Educational Institution of Higher Education «MIREA — Russian Technological University»

109383, Russia, Moscow region, Moscow, Polbina str., 35k2, 912

frit_027@mail.ru

Abstract. In the modern world, teamwork is becoming more and more common. Different participants may be in different places, but they still need to work together on the same project, including graphic diagrams. An important aspect of this approach is the ability to observe changes made by other participants in real time. This allows, first of all, to reduce the frequency of conflicts when simultaneously editing the same schema element. However, existing solutions for sharing data in real-time collaborative editing of graphical diagrams face a number of problems, such as delays in data transmission. The subject of research in this article is the development of a minimum viable web application that allows users to perform collaborative graphical editing of a canvas in real time. The object of the study is a model of the process of collaborative editing in real time, taking into account the resolution of emerging conflicts. The research methodology is based on a theoretical approach to identifying mathematical formulas that describe changes in the state of a document when it is jointly edited by users. The characteristics of the use of the HTTP and WebSocket protocols in multi-user client-server applications are given. To use the WebSocket protocol, the Socket.IO library is used. The application server is built using the Express framework. The authors' main contribution to the topic is a model of the real-time collaborative editing process, as well as a mechanism for detecting conflicts for any number of users and a conflict resolution function for each pair of conflicting changes when online collaborative editing of documents. Within the framework of this study, an algorithm for collaborative editing of graphic schemes in real time is additionally proposed and its implementation in the form of a software system is given. The algorithm proposed as a result of the study in the JavaScript programming language can be used as a basis for developing more rich web applications using the Socket.IO library and be the object of future research affecting multi-user interaction and real-time conflict resolution.

Keywords: algorithm, graphic scheme, JavaScript programming language, conflict resolution, conflict detection, collaborative editing, client-server application, WebSocket protocol, HTTP protocol, event driven

References (transliterated)

1. Knyazev A. A., Kondrat'ev A. N., Dubrovskii N. S. Evolyutsiya i osobennosti protokola HTTP // Innovatsionnyi potentsial razvitiya obshchestva: vzglyad molodykh uchenykh: sbornik nauchnykh statei 4-i Vserossiiskoi nauchnoi konferentsii perspektivnykh razrabotok, Kursk, 01 dekabrya 2023 goda. – Kursk: ZAO «Universitetskaya kniga», 2023. – S. 176-178.
2. Kovaliuk, D., Kovaliuk, O.O., Pinaieva, O., Kotyra, A., & Kalizhanova, A. (2019). Optimization of web-application performance. *Symposium on Photonics Applications in Astronomy, Communications, Industry, and High-Energy Physics Experiments (WILGA)*.
3. Gursesli, M.C.; Selek, M.E.; Samur, M.O.; Duradoni, M.; Park, K.; Guazzini, A.; Lanatà, A. Design of Cloud-Based Real-Time Eye-Tracking Monitoring and Storage System. Algorithms 2023, 16, 355. https://doi.org/10.3390/a16070355
4. Gorchakov A. Ya. Razrabotka klientskoi arkhitektury sistemy mgnovennykh soobshchenii po tekhnologii WebSocket // Gagarinskie chteniya-2018: Sbornik tezisov

dokladov XLIV Mezhdunarodnoi molodezhnoi nauchnoi konferentsii, Moskva-Baikonur-Akhtubinsk, 17–20 aprelya 2018 goda. Tom 2. – Moskva-Baikonur-Akhtubinsk: Moskovskii aviatsionnyi institut (natsional'nyi issledovatel'skii universitet), 2018. – S. 209.

5. Shabanov A. E. Obzor biblioteki socket.io // Informatsionno-komp'yuternye tekhnologii v ekonomike, obrazovanii i sotsial'noi sfere. – 2022. – № 1(35). – S. 56-62.
6. Tsareva E. V. Razreshenie konfliktnykh situatsii pri sinkhronizatsii mnogopol'zovatel'skikh online-prilozhenii // Vestnik Tomskogo gosudarstvennogo universiteta. Upravlenie, vychislitel'naya tekhnika i informatika. – 2016. – № 1(34). – S. 79-91.
7. Chernyshev Ya. R. Razrabotka i samostoyatel'nyi khosting veb-prilozheniya na osnove freimvorka Express.js // Studencheskaya nauka-vzglyad v budushchee: Materialy KhVIII Vserossiiskoi studencheskoi nauchnoi konferentsii, Krasnoyarsk, 15–17 marta 2023 goda. Tom Chast' 5. – Krasnoyarsk: Krasnoyarskii gosudarstvennyi agrarnyi universitet, 2023. – S. 291-293.
8. Karam, Sameer & Abdulrahman, Bikhtiyar. (2022). Using Socket.io Approach for Many-to-Many Bi-Directional Video Conferencing. AL-Rafidain Journal of Computer Sciences and Mathematics. 16. 81-86. 10.33899/csmj.2022.174411.
9. Macklon, Finlay & Viggiato, Markos & Romanova, Natalia & Buzon, Chris & Paas, Dale & Bezemer, Cor-Paul. (2023). A Taxonomy of Testable HTML5 Canvas Issues. IEEE Transactions on Software Engineering. PP. 1-13. 10.1109/TSE.2023.3270740.
10. Kochitov M. E. Risovanie komp'yuternoi mysh'yu na kholste veb-stranitsy brauzera s pomoshch'yu instrumenta Canvas // Postulat. – 2019. – № 8(46). – S. 25.

Development and application of operating systems and shells in mobile technologies: analysis of the history of development and current trends in the field of mobile OS and shells

Malakhov Sergei Valer'evich

PhD in Technical Science

Associate Professor, Department of Management in Technical Systems, Povolzhskiy State University of Telecommunications & Informatics

23 Lva Tolstogo str., Samara, Samara region, 443010, Russia

s.malakhov@psuti.ru

Yakupov Denis Olegovich

Assistant, Graduate student, Department of Management in Technical Systems, Povolzhskiy State University of Telecommunications & Informatics

23 Lva Tolstogo str., Samara, Samara region, 443010, Russia

d.yakupov@psuti.ru

Vorobeva Evgeniia Grigor'evna

Student, Department Informatics and computer technology, Povolzhskiy State University of Telecommunications & Informatics

443010, Russia, Samara region, Samara, L. Tolstogo str., 23

vorobeva.g2004@gmail.com

Nekhaev Maksim Vadimovich

Student, Department Informatics and computer technology, Povolzhskiy State University of Telecommunications & Informatics

443010, Russia, Samara region, Samara, L. Tolstogo str., 23

maks.popovich2014@yandex.ru

Muhtulov Mihail Olegovich

Student, Department Informatics and computer technology, Povolzhskiy State University of Telecommunications & Informatics

443010, Russia, Samara region, Samara, L. Tolstogo str., 23

mixa.1204@inbox.ru

Novoseltseva Sofiia Vladimirovna

Student, Department Informatics and computer technology, Povolzhskiy State University of Telecommunications & Informatics

443010, Russia, Samara region, Samara, L. Tolstogo str., 23

sunny.tea.with.lilac@gmail.com

Abstract. The objects of research are mobile operating systems and their shells. The subject of the study is the functionality of the Android, iOS and HarmonyOS operating systems, their history of creation and development trends. The authors consider in detail such aspects of the topic as the history of the creation of Android, iOS, HarmonyOS operating systems and TouchWiz, HTC Sense, MIUI and others shells, modern trends in the field of mobile operating systems, which reflect the influence of technological innovations and geopolitical aspects on the development of this sphere. They conduct a detailed analysis of the OS using performance testing applications (AnTuTu Benchmark, 3DMark Benchmark). The purpose of this research is to study the history of the development of mobile OS and shells from the origins to the current trends of technological progress. Research methods are based on the collection and systematization of information, analysis and comparison of systems, as well as performance tests. The scientific novelty of this article lies in the use of OS and shells in mobile devices that meet all the needs and requirements of the user, given the rapid development of digital technologies and their increasing introduction into our daily lives. The main conclusions of the study are the identification of the most common OS, the definition of modern trends, which include the integration of artificial intelligence, multimodality, security and privacy, as well as the expansion of flexibility and portability. The rapid development of technology and the universe of mobile applications makes mobile OS and shells key components of a successful user experience in the world of mobile technology. Understanding the history and current trends in the field of mobile operating systems and shells will allow to more accurately predict technological changes and potential impacts in the future.

Keywords: world market, artificial intelligence, operating, open source code, system, device, shell, mobile OS, operating system, software

References (transliterated)

1. Akhmetov A. K. Operatsionnaya sistema Android: istoriya sozdaniya i razvitiya. Razrabotka prilozhenii dlya platformy Android // Skif. Voprosy studencheskoi nauki. 2017. №9. S. 2-3.

2. Pogorelov D. V., Kolokolov E. A., Ermolaev V. V. Sravnenie mobil'nykh operatsionnykh sistem Android i ios // Vestnik nauki. 2022. №12 (57) T. 5. S. 120-124.
3. Staroverova N. A., Morozov D., Kalaev I., Kadyrova G. Sovremennye tendentsii i perspektivy razvitiya operatsionnykh sistem // Vestnik Kazanskogo tekhnologicheskogo universiteta. 2015. T. 18. №21. S. 134-136.
4. Sravnivaem luchshie operatsionnye sistemy dlya smartfonov v 2023 godu [Elektronnyi resurs]. URL: https://blog.eldorado.ru/publications/battl-os-sravnivaem-luchshie-operatsionnye-sistemy-dlya-smartfonov-v-2023-godu-39758 Data obrashcheniya: 10.03.2024.
5. AnTuTu Benchmark [Elektronnyi resurs]. URL: https://www.antutu.com/en/download.htm Data obrashcheniya: 10.03.2024.
6. ZDMark [Elektronnyi resurs]. URL: https://benchmarks.ul.com/3dmark-android Data obrashcheniya: 10.03.2024

Instrumental approach to programming in MultiOberon system

Dagaev Dmitry Viktorovich

General Director, SCADI Llc

115230, Russia, Moscow, Zeleny Prospekt, 5/12, building 3, room 1

dvdagaev@oberon.org

Abstract. Object-oriented approaches to programming have their own scope of applicability. For a number of tasks, preference is traditionally given to classical methods of structured programming. These preferences are not uncommon in a deterministic world and in machine-representation-oriented systems. Historically, classical methods developed from von Neumann's architecture of machine representation.
While solving the problems of deterministic world the advantage of approaches, opposite to OOP is shown. For example, the Oberon modular language and system in classic distribution demonstrate minimalistic way of reliability, which differs from vast majority of program systems maximizing amount of features supported. Data-oriented programming technology also steps aside traditional object-oriented paradigm because data from code separation is needed. The instrumental approach proposed by author is linking Oberon technologies with data-oriented programming, keeping interface interaction mechanisms from OOP. The instrument with no data, but associated with data is introduced instead of an object. MultiOberon restrictive semantics makes an opportunity to turn off OOP restriction and switch on instruments usage. Instrument is instantiated automatically on program module loading. Instrument is queried either by its type or by the type of record associated. All the functionality is implemented in MultiOberon compiler. Instrumental approach was used for SCADA-platform software development, which targets complex automation and diagnostics. It is used in dynamically loaded plugins for data types matched shared memory data types.
The instrumental approach offers a different branch of development from OOP for the classic Oberon programming language and the classical approach

Keywords: modularity, data oriented programming, MultiOberon, Metadata, Informatika-21, SCADA, instrumental approach, compiler, restriction, Oberon

References (transliterated)

1. David West. Object Thinking. Microsoft Press, 2004. S. 87-89.
2. Virt N., Algoritmy i struktury dannykh. DMK-Press, 2016. S. 272.
3. N. Virt, Yu. Gutknekht. Razrabotka operatsionnoi sistemy i kompilyatora. Proekt Oberon. DMK-Press, 2017. S. 560.
4. Dagaev D.V. Ogranichitel'naya semantika yazyka v sisteme Mul'tiOberon // Programmnye sistemy i vychislitel'nye metody. – 2023. – № 1. – S. 26-41. DOI: 10.7256/2454-0714.2023.1.36217 EDN: IWIODR
5. Rajive J., Ph.D. Data-Oriented Architecture: A Loosely-Coupled Real-Time SOA, Real-Time Innovations, Inc., 2007 August. S. 19-23.
6. Gamma E., Khelm R., Dzhonson R., Vlissides Dzh. Priemy ob''ektno-orientirovannogo proektirovaniya. Patterny proektirovaniya. SPb: Piter, 2019. S. 368.
7. Dagaev D.V. O razrabotke Oberon-sistemy s zadannymi svoistvami ergodichnosti. Trudy ISP RAN, tom 32, vyp. 6, 2020 g., str. 67-78. DOI: 10.15514/ISPRAS–2020–32(6)
8. Kopylov M.S., Deryabin N.B., Denisov E.Yu. Ob''ektno-orientirovannyi podkhod k podderzhke stsenariev v sistemakh opticheskogo modelirovaniya // Trudy Instituta sistemnogo programmirovaniya RAN. 2023. No 35(2). str. 169-180.
9. Nazar N., Aleti A., Zheng Y. Feature-based software design pattern detection // Journal of Systems and Software, 2022, vol. 185, pp. 1-12.
10. Yu D., Zhang P., Yang J., Chen Z., Liu C., Chen J. Efficiently detecting structural design pattern instances based on ordered sequences // Journal of Systems and Software, 2018, vol. 142, pp. 35–56.
11. Lo S. K., Lu Q., Zhu L., Paik H.-Y., Xu X., Wang C. Architectural patterns for the design of federated learning systems // Journal of Systems and Software, 2022, vol. 191, p. 357.
12. Hosking A., Nystrom N., Cutts Q., Brahnmath K. Optimizing the read and write barriers for orthogonal persistence // Advances in Persistent Object Systems, Morrison, Jordan, and Atkinson (Eds.). Morgan Kaufmann, 1999. p. 11.
13. Lefort A. A Support for Persistent Memory in Java // Computer science. Institut Polytechnique de Paris, 2023. English. p. 10.
14. GOST R MEK 60880, Programmnoe obespechenie komp'yuternykh sistem, vypolnyayushchikh funktsii kategorii A // 2009. str. 220.
15. Tanenbaum A. Sovremennye operatsionnye sistemy // 4-e izd. – SPb.: Piter, 2015. Str. 100-101.
16. Dagaev D.V. Ispolnyayushchaya mashina avtomatnykh programm // Nauchno-tekhnicheskii vestnik informatsionnykh tekhnologii, mekhaniki i optiki. 2021. T. 21, № 4. S. 525-534.

Research and development of algorithms for the formation of an effective ensemble of convolutional neural networks for image classification

Bondarenko Valerii Aleksandrovich

Graduate student, Department of Information Technology and Mathematics, Sochi State University

354002, Russia, Krasnodar Territory, Sochi, Verkhnyaya Lysaya Gora str., 64

✉ valeriybbond@mail.ru

Popov Dmitrii Ivanovich

Doctor of Technical Science

Professor, Department of Information Technology and Mathematics, Sochi State University

354002, Russia, Krasnodar Territory, Sochi, Politechnicheskaya str., 7

damitry@mail.ru

Abstract. The object of the research is artificial neural networks (ANN) with convolutional architecture for image classification. The subject of the research is the study and development of algorithms for constructing ensembles of convolutional neural networks (SNS) in conditions of limited training sample. The aim of the study is to develop an algorithm for the formation of an effective model based on an ensemble of convolutional SNS using methods of averaging the results of each model, capable of avoiding overfitting in the process of improving the accuracy of the forecast and trained on a small amount of data, less than 10 thousand examples. As a basic network, an effective SNA architecture was developed as part of the ensemble, which showed good results as a single model. The article also examines methods for combining the results of ensemble models and provides recommendations for the formation of the SNA architecture. The research methods used are the theory of neural networks, the theory of machine learning, artificial intelligence, methods of algorithmization and programming of machine learning models, a comparative analysis of models based on different algorithms using classical ensembling with simple averaging and combining the results of basic algorithms in conditions of limited sampling, taking into account weighted average. The field of application of the obtained algorithm and model is medical diagnostics in medical institutions, sanatoriums during primary diagnostic admission, using the example of a research task, the model is trained to classify dermatological diseases according to input photographs. The novelty of the study lies in the development of an effective algorithm and image classification model based on an ensemble of convolutional NS that exceed the prediction accuracy of basic classifiers, the process of retraining an ensemble of classifiers with deep architecture on a small sample volume is investigated, from which conclusions are drawn on the design of an optimal network architecture and the choice of methods for combining the results of several basic classifiers. As a result of the research, an algorithm has been developed for the formation of an ensemble of SNS based on an effective basic architecture and weighted average averaging of the results of each model for the classification task of image recognition in conditions of limited sampling.

Keywords: pre-treatment, methods of neural network ensembling, medical diagnostics, the task of classification, ensembles of neural networks, weighted average, methods of averaging the results, convolutional architecture, artificial neural networks, balanced voting

References (transliterated)

1. Thoma M. Analysis and optimization of convolutional neural network architectures, 2017.
2. Cruz Y. J. et al. Ensemble of convolutional neural networks based on an evolutionary algorithm applied to an industrial welding process //Computers in Industry. – 2021. – T. 133. – S. 103-530.
3. Yang S. et al. An ensemble classification algorithm for convolutional neural network based on AdaBoost //2017 IEEE/ACIS 16th International Conference on Computer and Information Science (ICIS). – IEEE, 2017. – S. 401-406.
4. Basili V. R., Briand L. C., Melo W. L. A validation of object-oriented design metrics as

quality indicators //IEEE Transactions on software engineering. – 1996. – Т. 22. – №. 10. – S. 751-761.

5. Neironnye seti. Pereobuchenie-chto eto i kak etogo izbezhat', kriterii ostanova obucheniya. [Elektronnyi resurs]. URL: https://proproprogs.ru/neural_network/ pereobuchenie-chto-eto-i-kak-etogo-izbezhat-kriterii-ostanova-obucheniya (data obrashcheniya 09.02.2024).
6. Voronetskii Yu. O., Zhdanov N. A. Metody bor'by s pereobucheniem iskusstvennykh neironnykh setei // Nauchnyi aspekt. 2019. №2. [Elektronnyi resurs] URL: https://na-journal.ru/2-2019-tehnicheskie-nauki/1703-metody-borby-s-pereobucheniem-iskusstvennyh-neironnyh-setei (data obrashcheniya: 10.02.2024).
7. Li C. et al. Improving forecasting accuracy of daily enterprise electricity consumption using a random forest based on ensemble empirical mode decomposition // Energy. – 2018. – Т. 165. – S. 1220-1227.
8. Omisore O. M. et al. Weighting-based deep ensemble learning for recognition of interventionalists' hand motions during robot-assisted intravascular catheterization // IEEE Transactions on Human-Machine Systems. – 2022. – Т. 53. – №. 1. – S. 215-227.
9. Ansamblirovanie modelei neironnykh setei s ispol'zovaniem biblioteki Keras. [Elektronnyi resurs]. URL:https://se.moevm.info/lib/ exe/fetch.php/courses:artificial_ neural_ networks:pr_8.pdf (data obrashcheniya 11.02.2024).
10. Metod optimizatsii Neldera — Mida. Primer realizatsii na Python. [Elektronnyi resurs]. URL:https://habr.com/ru/articles/332092/ (data obrashcheniya 09.02.2024).
11. Klyueva I. A. Metody i algoritmy ansamblirovaniya i poiska znachenii parametrov klassifikatorov. [Elektronnyi resurs]. URL:https://dissov.pnzgu.ru/files/dissov .pnzgu.ru/2021/tech/klyueva/ dissertaciya_ klyuevoy _i_a_.pdf (data obrashcheniya 08.02.2024).
12. Mikryukov, A. A. Klassifikatsiya sobytii v sistemakh obespecheniya informatsionnoi bezopasnosti na osnove neirosetevykh tekhnologii / A. A. Mikryukov, A. V. Babash, V. A. Sizov // Otkrytoe obrazovanie. – 2019. – Т. 23. № 1. – С. 57-63.
13. Gizluk D. Adaptive optimization methods. // Neural networks are simple (part 7). 2020. №7. [Elektronnyi resurs]. URL:https://www.mql5.com/ru/articles/8598#para21 (data obrashcheniya: 10.02.2024).
14. Mason L. et al. Boosting algorithms as gradient descent //Advances in neural information processing systems. – 1999. – Т. 12.
15. Zaheer R., Shaziya H. A study of the optimization algorithms in deep learning //2019 third international conference on inventive systems and control (ICISC). – IEEE, 2019. – S. 536-539.
16. Staroverov B. A., Khamitov R. N. Realizatsiya glubokogo obucheniya dlya prognozirovaniya pri pomoshchi ansamblya neironnykh setei //Izvestiya Tul'skogo gosudarstvennogo universiteta. Tekhnicheskie nauki. – 2023. – №. 4. – S. 185-189.
17. Onan A., Korukoğlu S., Bulut H. A multiobjective weighted voting ensemble classifier based on differential evolution algorithm for text sentiment classification // Expert Systems with Applications. – 2016. – Т. 62. – S. 1-16.
18. Kim H. et al. A weight-adjusted voting algorithm for ensembles of classifiers //Journal of the Korean Statistical Society. – 2011. – Т. 40. – №. 4. – S. 437-449.
19. Yao X., Islam M. M. Evolving artificial neural network ensembles //IEEE Computational Intelligence Magazine. – 2008. – Т. 3. – №. 1. – S. 31-42.
20. Anand V. et al. Weighted Average Ensemble Deep Learning Model for Stratification of Brain Tumor in MRI Images //Diagnostics. – 2023. – Т. 13. – №. 7. – S. 1320.

21. The International Skin Imaging Collaboration. [Elektronnyi resurs].-URL: https://www.isic-archive.com (data obrashcheniya 12.02.2024).
22. Alexandropoulos S. A. N., Kotsiantis S. B., Vrahatis M. N. Data preprocessing in predictive data mining // The Knowledge Engineering Review. – 2019. – T. 34. – S. e1.
23. García S., Luengo J., Herrera F. Data preprocessing in data mining. – Cham, Switzerland: Springer International Publishing, 2015. – T. 72. – S. 59-139.
24. Liang G., Zheng L. A transfer learning method with deep residual network for pediatric pneumonia diagnosis // Computer methods and programs in biomedicine. – 2020. – T. 187. – S. 104-964.
25. InceptionV3. [Elektronnyi resurs].-URL: https://keras.io/api/ applications/inceptionv3/ (data obrashcheniya 13.02.2024).
26. InceptionResnNetV2. [Elektronnyi resurs]. URL: https://keras.io/api/ applications/inceptionresnetv2/ (data obrashcheniya 13.02.2024).
27. VGG16. [Elektronnyi resurs]. URL: https://keras.io/api/ applications/vgg/#vgg16-function (data obrashcheniya 13.02.2024).
28. Shchetinin E. Yu. O nekotorykh metodakh segmentatsii izobrazhenii s primeneniem svertochnykh neironnykh setei // Informatsionno-telekommunikatsionnye tekhnologii i matematicheskoe modelirovanie vysokotekhnologichnykh sistem. – 2021. – S. 507-510.
29. Rosebrock A. Change input shape dimensions for fine-tuning with Keras. // AI & Computer Vision Programming. 2019. [Elektronnyi resurs]. URL:https://pyimagesearch.com/2019/06/24/ change-input-shape-dimensions-for-fine-tuning-with-keras/ (data obrashcheniya 14.02.2024).
30. Kostin K. A. i dr. Adaptivnyi klassifikator patologii dlya komp'yuternoi diagnostiki zabolevanii s ispol'zovaniem svertochnykh neironnykh setei po meditsinskim izobrazheniyam i videodannym: magisterskaya dissertatsiya po napravleniyu podgotovki: 01.04. 02-Prikladnaya matematika i informatika. – 2017.
31. A. Krizhevsky, I. Sutskever, G.E. Hinton. ImageNet Classification with Deep Convolutional Neural Networks. Proceedings of Advances in Neural Information Processing Systems 25 (NIPS 2012), 2012, Pp. 1097-1105.
32. Wang J., Lin J., Wang Z. Efficient hardware architectures for deep convolutional neural network // IEEE Transactions on Circuits and Systems I: Regular Papers. – 2017. – T. 65. – №. 6. – pp. 1941-1953.
33. Phung V. H., Rhee E. J. A high-accuracy model average ensemble of convolutional neural networks for classification of cloud image patches on small datasets //Applied Sciences. – 2019. – T. 9. – №. 21. – S. 4500.
34. The differential evolution method. [Elektronnyi resurs]. URL: https://docs.scipy.org/ doc/scipy/reference/generated/scipy.optimize.differential_evolution.html (data obrashcheniya: 13.02.2024).
35. Kak razrabotat' srednevzveshennyi ansambl' dlya glubokikh obuchayushchikh neironnykh setei. // 2018. [Elektronnyi resurs]. URL: https://machinelearningmastery.ru/ weighted-average-ensemble-for-deep-learning-neural-networks/# (data obrashcheniya: 13.02.2024)

Analysis of approaches to creating a «Smart Greenhouse» system based on a neural network

Cherepenin Valentin Anatolyevich

Postgraduate student, Department of Information and Measurement Systems and Technologies, South Russian State Polytechnic University (NPI) named after M.I. Platova

132 Prosveshcheniya str., Novocherkassk, Rostov region, 346428, Russia

cherept2@gmail.com

Katsupeev Andrei Aleksandrovich

PhD in Technical Science

Associate Professor, Department of Information and Measurement Systems and Technologies, South Russian State Polytechnic University (NPI) named after M.I. Platova

132 Prosveshcheniya str., Novocherkassk, Rostov region, 346428, Russia

andreykatsupeev@gmail.com

Abstract. The study addresses the crucial topic of designing and implementing smart systems in agricultural production, focusing on the development of a "Smart Greenhouse" utilizing neural networks. It thoroughly examines key technological innovations and their role in sustainable agriculture, emphasizing the collection, processing, and analysis of data to enhance plant growth conditions. The research highlights the efficiency of resource use, management of humidity, temperature, carbon dioxide levels, and lighting, as well as the automation of irrigation and fertilization. Special attention is given to developing adaptive algorithms for predicting optimal conditions that increase crop yield and quality while reducing environmental impact and costs. This opens new avenues for the sustainable development of the agricultural sector, promoting more efficient and environmentally friendly farming practices. Utilizing a literature review, comparative analysis of existing solutions, and neural network simulations for predicting optimal growing conditions, the study makes a significant contribution to applying artificial intelligence for greenhouse microclimate management. It explores the potential of AI in predicting and optimizing growing conditions, potentially leading to revolutionary changes in agriculture. The research identifies scientific innovations, including the development and testing of predictive algorithms that adapt to changing external conditions, maximizing productivity with minimal resource expenditure. The findings emphasize the importance of further studying and implementing smart systems in agriculture, highlighting their potential to increase yield and improve product quality while reducing environmental impact. In conclusion, the article assesses the prospects of neural networks in the agricultural sector and explores possible directions for the further development of "Smart Greenhouses".

Keywords: neural network, optimization algorithm, microclimate, hybrid neural network, deep learning, biotechnology, greenhouse, internet of things, Smart Greenhouse, Analysis of approaches

References (transliterated)

1. Taki, M. "Solar thermal modeling and application in greenhouses"/ M. Taki, A. Rohani, M. Rahmati-Joneidabad // Info Proc Agri, 5 (2018), pp. 83-113. DOI:10.1016/J.inpa.2017.10.003.
2. Huihui, Yu "Temperature prediction in a Chinese solar greenhouse based on LS-SVM optimized with improved PSO" / Huihui Yu, Yinyi Chen, Shahbaz Gul Hassan, Daoliang Li// Comput Electron Agric, 122 (2016), pp. 94-102. DOI:10.1016/J.compag.2016.01.019.

3. Taki, M. "The relationship of energy and yield costs and sensitivity analysis for growing tomatoes in greenhouses in Iran" / M.Taki, R.Abdi, M.Akbarpour, H.Ghasemi-Mobtaker // Agric Eng Int: CIGR J, 15 (2013), pp. 59-67. ISSN:16821130.
4. Abdel Ghani, A.M. "The use of solar energy by greenhouses: general relations" / A.M. Abdel Ghani, I.M. Helal // Renewable Energy, 36 (2011), pp. 189-196. DOI:10.1016/j.renene.2010.06.020.
5. Nielsen, H. "Identification of transfer functions for regulating air temperature in a greenhouse" / H.Nielsen, P.Madsen // J Agric Eng Res, 60 (1995), pp. 25-34. DOI:10.1006/jaer.1995.1093.
6. Dariushi, E. "Labyrinth prediction of internal parameters of a tomato greenhouse in a semi-arid zone using a time series model of artificial neural networks" / E. Dariushi, K. Aassif, L. Lekush, G. Buirden // Measurement, 42 (2009), pp. 456-463. DOI:10.1016/J.measurement.2008.08.013
7. Katsupeev, A. A. Postanovka i formalizatsiya zadachi formirovaniya informatsionnoi zashchity raspredelennykh sistem / A. A. Katsupeev, E. A. Shcherbakova, S. P. Vorob'ev // Inzhenernyi vestnik Dona. – 2015, 34, c. 21. – EDNTXTMHJ.
8. Abdi, R. "Analysis of energy consumption and greenhouse gas emissions from agricultural production" / R. Abdi, M. Taki, M. Akbarpour // Int J Nat Eng Sci,6 (2012), pp. 73-79. ISSN: 026322410.
9. Ruano, A.E. "Forecasting building temperature using neural network models" / A.E. Ruano, E.M. Crispim, E.Z.E. Conceicao, M.Lucio // Energy Build, 38 (2006), pp. 682-694. DOI:10.1016/J.enbuild.
10. He, F. "Modeling of air humidity in a greenhouse using an artificial neural network and analysis of the main components" / F. He, S. Ma // Prog Electron-X. Sciences, 71 (2010), pp. 19-23. DOI: 10.1016/J.compag.2009.07 0.011.
11. Taki, M. "Models of heat transfer and MLP neural networks for predicting internal environmental variables and energy losses in a semi – solar greenhouse" / M. Taki, Yu. Ajabshirchi, S. F. Ranjbar, A. Rohani, M. Matlub // Energy Build, 110 (2016), pp. 314-329. DOI:10.1016/j.enbuild.2015.11.010.
12. Ilyas, S.A. "Neural network logic sensor RBF for monitoring technological emissions" / Ilyas S.A., Elshafey M., Habib M.A. // Control Eng Practice, 21 (2013), pp. 962-970. DOI:10.1016/J.conengprac.2013.01.007.
13. Vorob'ev, S. P. Issledovanie modeli tranzaktsionnoi sistemy s replikatsiei fragmentov bazy dannykh, postroennoi po printsipam oblachnoi sredy / S. P. Vorob'ev, V. V. Gorobets // Inzhenernyi vestnik Dona, 2012, 22, s. 49. – EDNPRXKMH