ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

На правах рукописи

Ширяев Егор Михайлович

Математическая модель, методы и алгоритмы эффективной реализации искусственных нейронных сетей, сохраняющих конфиденциальность

Специальность 2.3.5 —
«Математическое и программное обеспечение вычислительных систем, комплексов и компьютерных сетей»

Диссертация на соискание ученой степени
кандидата физико-математических наук

Научный руководитель:
Доктор физико-математических наук, доцент
Бабенко Михаил Григорьевич

Ставрополь — 2025

Оглавление

Стр.

Стр.

Стр.

Введение

Актуальность темы исследования. Развитие методов искусственного интеллекта, а также вычислительной аппаратуры привело к тому, что машинное обучение и искусственные нейронные сети используются уже не только для узконаправленных специализированных задач, но и для широкого круга задач различных сфер деятельности человека. Особое внимание стоит уделить методам анализа Больших данных, отметить их быстрое развитие и повсеместное внедрение в производство для аналитического сопровождения различных процессов. С развитием методов машинного обучения и искусственных нейронных сетей растет и вычислительная сложность решаемых с их помощью задач, что приводит к дефициту вычислительных мощностей, как у рядового пользователя, так и у малого и среднего бизнеса.

Наиболее рациональным способом снижения затрат на необходимые вычислительные ресурсы является применение облачных технологий, а именно услуг облачных провайдеров. Однако в данном случае возникают определенные риски, связанные с конфиденциальностью обрабатываемых данных. В целом риски можно разделить на две категории:

1. Внешние риски.
2. Внутренние риски.

К внешним рискам можно отнести любого рода кибератаки на безопасность системы облачного провайдера. В данном случае угрозы конфиденциальности информации заключаются в том, что злоумышленник может украсть информацию, которая в настоящий момент обрабатывается в облаке, либо повредить ее, нарушив целостность и/или внеся в нее изменения. К внутренним рискам обычно относятся случаи, когда злоумышленник либо внедрился в персонал облачного провайдера, либо имеет доступ к системе администрирования, либо существует предварительный сговор с персоналом облачного провайдера. Таким образом, в случае использования облачных сервисов, наряду со стандартными рисками нарушения конфиденциальности возникают дополнительные, связанные с удаленной обработкой конфиденциальных данных.

Вопрос конфиденциальности данных, обрабатываемых методами искусственного интеллекта в облачных вычислительных системах, в настоящий момент становится критическим. Такая ситуация складывается в связи с

внедрением методов машинного обучения и искусственных нейронных сетей в чувствительные сферы деятельности человека, такие как, например, здравоохранение, финансовый сектор, государственный сектор. Здесь конфиденциальными данными могут выступать персональные данные пользователя, данные составляющие врачебную тайну, корпоративную тайну и в некоторых случаях государственную тайну. Кроме того, если рассматривать такие концепции, как «Умный город», то вопрос конфиденциальности данных становится еще более острым, так как возникают риски для данных достаточного широкого спектра.

Сложившаяся ситуация осложняется тем, что в настоящий момент на территории Российской Федерации одним из приоритетов является переход к передовым цифровым интеллектуальным производственным технологиям, роботизированным системам, новым материалам и способам конструирования, созданию систем обработки больших объемов данных, машинного обучения и искусственного интеллекта. Что так же влечет за собой повышение внимания к вопросу конфиденциальности данных, обрабатываемых в облачных системах, в том числе с применением аппарата искусственных нейронных сетей.

Искусственные нейронные сети находят все большее применение в различных областях человеческой деятельности. В здравоохранении они применяются для распознавания и анализа результатов исследований пациентов. В финансовом секторе – для анализа курсов валют и прогнозирования роста/снижения стоимости отдельных активов и биржи в целом. На муниципальном уровне – для анализа трафика на дорогах, потребления коммунальных услуг и т.п. Корпорации применяют искусственные нейронные сети для анализа и оптимизации внутренних процессов. Рядовые пользователи используют искусственные нейронные сети для личных нужд. Если в процессе обучения нейронных сетей данные являются открытыми и общедоступными, то данные, которые подаются на вход обученной нейронной сети, зачастую являются конфиденциальными.

В целях обеспечения конфиденциальности обрабатываемых искусственными нейронными сетями данных требуется разработка методов и алгоритмов, реализующих функционал криптографических систем, но с возможностью корректно дешифровать результат после обработки зашифрованных данных. В данной области наиболее перспективным направлением является полностью гомоморфное шифрование (ПГШ), оно позволяет обрабатывать зашифрованные данные с помощью гомоморфного сложения и умножения. В контексте

искусственных нейронных сетей операции сложения и умножения являются ключевыми, функции активации нейронов, хоть зачастую и построены на применении операции сравнения, могут быть приближенны с помощью численных методов. Такое мнение транслируют авторы работ [10; 141]. В своих исследованиях они демонстрируют перспективы применения ПГШ в области машинного обучения и нейронных сетей. Основными препятствиями для применения схем ПГШ являются высокая вычислительная сложность и ограниченность набора поддерживаемых операций, как правило сложением и умножением. Преодоление этих препятствий требует тщательного исследования, разработки и адаптации моделей, методов и алгоритмов, как гомоморфных операций ПГШ, так и самих искусственных нейронных сетей. Получение новых научных результатов в данной области исследований позволит повысить эффективность искусственных нейронных сетей, сохраняющих конфиденциальность, и расширить область их применения в первую очередь за счет расширения класса задач, допускающих применение облачных технологий.

Значительный научный вклад в рассматриваемую область внесли следующие исследователи: C. Gentry, V. Vaikuntanathan, Z. Brakerski, Yu. Polyakov, V. Shoup, A. Alexandru, A. Kim, S. Halevi, V. Zucca, D. Micciancio, J. H. Cheon, A. Al Badawi, Л. К. Бабенко, М. Г. Бабенко и М. А. Дерябин.

Целью диссертационного исследования является разработка методов и алгоритмов базовых операций искусственных нейронных сетей, сохраняющих конфиденциальность, уменьшающих время их обработки.

Объектом диссертационного исследования являются искусственные нейронные сети, сохраняющие конфиденциальность.

Предмет исследования – методы и алгоритмы искусственных нейронных сетей, сохраняющих конфиденциальность.

Научная задача диссертационной работы состоит в исследовании и разработке математической модели, методов и алгоритмов для повышения эффективности реализаций искусственных нейронных сетей, сохраняющих конфиденциальность на основе схем полностью гомоморфного шифрования.

Для решения поставленной общей научной задачи была произведена ее декомпозиция на ряд **частных задач**:

1. Разработка методов и алгоритмов для проектирования слоев искусственной нейронной сети, сохраняющей конфиденциальность.

2. Адаптация существующих моделей искусственных нейронных сетей под ограничения гомоморфных шифров.
3. Разработка комплекса программ, реализующего функционал искусственных нейронных сетей на базе схем гомоморфного шифрования.

Соответствие паспорту научной специальности. Область исследования соответствует паспорту специальности 2.3.5 – «Математическое и программное обеспечение вычислительных систем, комплексов и компьютерных сетей» по следующим пунктам:

п. 3. Модели, методы, архитектуры, алгоритмы, языки и программные инструменты организации взаимодействия программ и программных систем.

п. 4. Интеллектуальные системы машинного обучения, управления базами данных и знаний, инструментальные средства разработки цифровых продуктов.

п. 9. Модели, методы, алгоритмы, облачные технологии и программная инфраструктура организации глобально распределенной обработки данных.

Научная новизна:

1. Предложен модифицированный метод умножения матриц для реализации слоев искусственной нейронной сети, сохраняющей конфиденциальность, позволяющий уменьшить вычислительную сложность алгоритма с $O(n^4)$ до $O(n^2)$.
2. Разработаны методы дистилляции и квантизации параметров для адаптации искусственных нейронных сетей к ограничениям гомоморфных шифров, позволяющие сократить потребление памяти примерно в 1500 раз, уменьшить время обработки данных в среднем в 30 раз, при потере доли верных классификаций в диапазоне от 0.5% до 1%.
3. Разработаны функции активации для проектирования искусственных нейронных сетей, сохраняющих конфиденциальность.
4. Разработан программный комплекс для реализации искусственных нейронной сети, сохраняющей конфиденциальность.

Моделирование и вычислительный эксперимент проведены на базе платформы Yandex Cloud в сервисе DataSphere на конфигурации c1.32, которая включает 32 виртуальных центральных процессора Intel Ice Lake и 256 ГБ оперативной памяти, с использованием языка программирования высокого уровня Python для разработки модулей, библиотеки PyTorch для проектирования от-

крытых искусственных нейронных сетей и библиотек TenSEAL и Concrete-ML для проектирования искусственных нейронных сетей, сохраняющих конфиденциальность. В качестве критериев оценки эффективности разработанных моделей, методов и алгоритмов, использовались время одной итерации работы искусственной нейронной сети (миллисекунды, мс и секунды, с); память, занимаемая искусственной нейронной сетью (гигабайты, ГБ); точность распознавания 10 000 изображений искусственной нейронной сетью (проценты, %).

Практическая значимость разработанных математической модели, методов и алгоритмов заключается в возможности реализации на их основе моделей искусственных нейронных сетей, сохраняющих конфиденциальность, для обработки входных данных в зашифрованном виде. Полученные результаты могут быть использованы при построении моделей искусственных нейронных сетей в областях с повышенными требованиями к конфиденциальности данных, как, например, здравоохранение, финансовый сектор, банковская деятельность и т.п.

Методология и методы исследования включают использование математического аппарата линейной алгебры, математического анализа, теории алгоритмов, численных методов, математического моделирования, теории распознавания образов и математической статистики.

Основные положения, выносимые на защиту:

1. Метод умножения зашифрованных матриц.
2. Алгоритмы проектирования сверточного и скрытого слоев нейронной сети на базе модифицированного метода умножения матриц, сохраняющего конфиденциальность.
3. Методы адаптации искусственных нейронных сетей учитывающие ограничения гомоморфных шифров.
4. Алгоритмы полиномиального приближения стандартных функций активации и функции активации на основе полиномов с обучаемыми коэффициентами.

Основные результаты диссертационного исследования были использованы в рамках следующих научно-технических работ:

1. РНФ № 19-71-10033 «Эффективная, безопасная и отказоустойчивая система распределенного хранения и обработки конфиденциальных данных с регулируемой избыточностью для проектирования мобильных облаков на маломощных вычислительных устройствах»;

2. РНФ № 22-71-10046 «Разработка новых методов и алгоритмов для повышения надежности и безопасности хранения, передачи и обработки данных в туманных вычислениях»;
3. Северо-Кавказский федеральный университет «Интеллектуальный блок управления распределенной системой хранения данных в гетерогенных средах с регулируемой избыточностью и безопасностью»;
4. РНФ № 25-71-30007 «Новые технологии для проектирования облачных сервисов машинного обучения, сохраняющих конфиденциальность» (2 и 3 глава).

Достоверность полученных результатов обеспечивается строгостью проведения математических доказательств, при получении которых был использован научно-методический аппарат математического анализа, теории чисел и численных методов, и подтверждается проведенным сравнительным анализом разработанных методов и алгоритмов с известными ранее с точки зрения скорости обработки данных, потребления оперативной памяти и точности распознавания образов.

Личный вклад автора. Все изложенные в диссертационной работе результаты получены при непосредственном участии автора. Из результатов работ, выполненных коллективно, в диссертацию включены только полученные непосредственно автором. В работах [14—16; 25; 28; 112; 129; 147; 160; 161] автором рассмотрены модели распределенных вычислений, а именно облачных и туманных вычислений, проанализированы и обозначены их уязвимости с точки зрения безопасности. В работах [67; 165] автором проанализирована безопасность «умных городов», уточнены требования к безопасности. В работах [128; 131] автором исследованы методы машинного обучения и искусственных нейронных сетей. В работах [1; 3; 8; 17; 70; 87; 159] исследованы вычислительные характеристики гомоморфных шифров, а также методы повышения их эффективности. В работах [11; 31; 38; 96] автором проведены исследования искусственных нейронных сетей, сохраняющих конфиденциальность, на базе гомоморфных шифров, разработан метод умножения матриц, сохраняющий конфиденциальность входных данных, характеризующийся меньшим потреблением памяти и меньшей вычислительной сложностью, достигаемыми путем сокращения количества гомоморфных операций. Разработан комплекс программ для разработки и исследования искусственных нейронных сетей, сохраняющих конфиденциальность, с применением схем ПГШ [4—6].

Апробация работы. Основные результаты диссертационного исследования докладывались на международных конференциях, среди которых «International Workshop on Advanced in Information Security Management and Applications (AISMA-2023)» (г. Алигарх, Индия, г. Ставрополь, г. Красноярск, Россия), «Conference on Current Problems of Applied Mathematics and Computer Systems (CPAMCS 2023)» (г. Ставрополь, Россия), «V Ibero-American Congress of Smart Cities (ICSC-CITIES 2022)» (г. Куэнка, Эквадор), «International Conference on Mathematics and its Applications in new Computer Systems (MANCS 2021)» (г. Ставрополь, Россия), «2021 IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering (ElConRus 2021)» (г. Москва и г. Санкт-Петербург, Россия), «Advances in Automation II: Proceedings of the International Russian Automation Conference (RusAutoConf 2020)» (г. Сочи, Россия), «The International Workshop on Information, Computation, and Control Systems for Distributed Environments» (г. Иркутск, Россия), «Spring/Summer Young Researchers' Colloquium on Software Engineering (SYRCoSE 2020, 2024)» (г. Ставрополь, Россия), «International Conference Engineering and Telecommunication (En&T 2021)» (г. Москва, Россия), «IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW 2021)» (г. Портленд, Орегон, США), «International Conference on High Performance Computing & Simulation (HPCS 2019)» (г. Дублин, Ирландия).

Публикации. Основные результаты по теме диссертационного исследования изложены в 29 публикациях [1; 3—6; 8; 11; 12; 14—17; 25; 28; 31; 38; 67; 70; 87; 96; 112; 128; 129; 131; 147; 159—161; 165], 4 из которых изданы в журналах, рекомендованных ВАК [1; 25; 31; 96], 15 – в тезисах докладов конференций [14—16; 28; 38; 70; 87; 112; 128; 129; 131; 159—161; 165], 8 – в журналах, входящих в международные базы цитирования Web of Science и Scopus [11; 12; 17; 25; 31; 67; 96; 147]. Получено 3 свидетельства о государственной регистрации программ для ЭВМ [4—6].

Объем и структура работы. Диссертация состоит из введения, 3 глав, заключения.

В первой главе рассмотрены распределенные вычислительные системы, а именно облачные и туманные вычисления. Проведен анализ их безопасности, выявлены проблемы конфиденциальности таких систем. Рассмотрены методы искусственного интеллекта, особое внимание уделено искусственным нейронным сетям, а именно классу сверточных нейронных сетей. Проанализировано

применение методов искусственного интеллекта в публичных распределенных системах. В качестве методов обеспечения конфиденциальности выбраны гомоморфные шифры, среди которых были выделены три основных схемы: TFHE, CKKS и BFV.

Во второй главе представлено исследование библиотек, программно реализующих схемы гомоморфного шифрования. Определены две библиотеки, наиболее подходящие для проведения дальнейших исследований: TenSEAL и Concrete-ML. Данные библиотеки реализованы на языке высокого уровня Python, предоставляют инструментарий для работы с гомоморфными шифрами и их внедрения в искусственные нейронные сети. Исследованы методы матричного умножения, сохраняющего конфиденциальность. Сформулированы и доказаны теоремы об умножении зашифрованной матрицы на открытую и об аппроксимации функции искусственной нейронной сетью. На основе полученной теоремы предложена модификация гомоморфного умножения матрицы зашифрованных конфиденциальных входных данных на открытую матрицу параметров, что позволило сократить объем памяти в среднем в 7.89 раза, уменьшить пространственную сложность с $O(n^4)$ до $O(n^2)$ для произведения матриц размера $n \times n$ и сократить вычислительную сложность в среднем в 1.49 раза.

Проведено исследование методов дистилляции сверточных нейронных сетей (СНС), модифицирован метод дистилляции для задач классификации, который позволяет проводить дистилляцию для задач распознавания образов, определены оптимальные константа контроля ошибки, равная $a \approx 0.875$, и размер коалиции сверточных нейронных сетей учителей, равный 6, что позволило уменьшить размеры СНС более чем в 1609 раз при потере точности $\pm$0.5-1%.

Проведено исследование различных методов квантизации на базе схем гомоморфного шифрования CKKS и BFV. Исследование показало, что метод квантизации 2^N дает наилучшее соотношение скорости обработки данных и точности распознавания изображений. Кроме того, исследование показало, что схема BFV проигрывает схеме CKKS и в точности распознавания изображений, и по скорости обработки данных.

В третьей главе разработаны математические модели сверточных нейронных сетей, как открытых, так и обеспечивающих конфиденциальность, на базе нескольких приближенных функций активации. Это необходимо для более

детального исследования моделей сверточных нейронных сетей сохраняющих конфиденциальность на базе гомоморфных схем CKKS и TFHE.

Построенные математические модели были исследованы в два этапа и на двух наборах данных. Первый этап – исследование моделей сверточных нейронных сетей сохраняющих конфиденциальность без применения дистилляции, второй этап – с применением дистилляции. В качестве наборов данных были выбраны MNIST с изображениями размером 28×28 и 8×8. Для этого были построены модели СНС-учителей и модели СНС-учеников; получены полиноминальные аппроксимации различных функций активации; для функции активации с обучаемыми коэффициентами в ходе обучения СНС были получены полиномы второй степени.

В результате исследования было установлено, что в большинстве случаев схема TFHE обрабатывает данные быстрее в среднем на 50%, однако CKKS имеет более высокую точность распознавания, в среднем на 5%.

Если рассматривать соотношение скорости обработки данных и точности распознавания изображений, то модели на базе схемы TFHE имеют наилучший результат при применении дистилляции и работе с наборами данных размера 8×8, модели на базе CKKS напротив дают наилучший результат при распознавании изображений размера 28×28. Получены результаты применения приближенных функций активации: модели на базе CKKS показывают наилучшие результаты, в то время как модели на базе TFHE зачастую не могут обеспечить точность выше 90% на заданных параметрах. Лучший результат распознавания был получен на основе функции активации с обучаемыми коэффициентами в моделях на базе CKKS. Таким образом, в ходе проведения исследования были получены оптимальные модели сверточных нейронных сетей, сохраняющих конфиденциальность, для различных наборов данных с применением группы функций активации, обеспечивающих требуемое соотношение точности и скорости обработки данных. Полученные в ходе исследования результаты были использованы при разработке программного комплекса для проектирования сверточных нейронных сетей, сохраняющих конфиденциальность.

Разработанный программный комплекс потенциально позволит расширить применение моделей СНС за счет возможности обработки конфиденциальных данных в общедоступных облаках и туманных вычислениях.

Полный объём диссертации составляет 156 страниц, включая 39 рисунков и 6 таблиц. Список литературы содержит 186 наименований.

Глава 1. Анализ проблемы конфиденциальности искусственного интеллекта в публичных распределенных системах

1.1 Применение искусственного интеллекта в распределенных вычислительных системах

1.1.1 Виды распределенных вычислительных систем

Распределенные вычислительные системы (РВС) быстро развивались на протяжении нескольких последних десятилетий, что было обусловлено потребностью в эффективных и масштабируемых вычислительных моделях [163]. Начиная с 1960-х годов, вопрос о распределенных вычислениях начал подниматься все чаще, когда исследователи начали изучать концепцию разделения времени, позволяющую нескольким пользователям одновременно получать доступ к одному и тому же компьютеру. Это заложило основу для разработки распределенных систем, в которых вычислительные ресурсы распределялись между несколькими узлами.

В 1970-х и 1980-х годах исследовательские усилия были сосредоточены на разработке локальных сетей (LAN – Local Area Network) и глобальных вычислительных сетей (WAN – Wide Area Network) для подключения географически распределенных компьютеров. Это привело к появлению архитектуры клиент-сервер, в которой центральный сервер обрабатывал запросы от нескольких клиентских устройств. Доминирующими стали РВС, использующие архитектуру клиент-сервер, позволяющие совместно использовать ресурсы и централизованно управлять данными.

Появление Интернета в 1990-х годах привело к значительным достижениям в области распределенных вычислений. Модель клиент-сервер расширилась, включив в себя веб-приложения, а Всемирная паутина (WWW – World Wide Web) стала платформой для распределенных вычислений. Эта эпоха ознаменовалась появлением веб-сервисов и интерфейсов прикладного программирования (API – Application Programming Interface), которые обеспечивали взаимодей-

ствие между различными системами, упрощая обмен данными и услугами через Интернет.

Облачные вычисления (ОВ), какими мы их знаем сегодня, появились в начале 2000-х годов и произвели революцию в способах предоставления, управления и использования вычислительных ресурсов [111]. Облачные технологии основаны на идее предоставления доступа по требованию к пулу вычислительных ресурсов, таких как вычислительные мощности, хранилища и программные приложения, поставляемых через Интернет.

Концепция облачных вычислений основана на более ранних разработках в области распределенных систем, виртуализации и сервисных вычислений. Технологии виртуализации позволяют абстрагироваться от физических ресурсов, предоставляя возможность создавать виртуальные машины (ВМ), которые можно динамически подготавливать и освобождать по мере необходимости.

Ключевые характеристики облачных вычислений включают самообслуживание по требованию, широкий доступ к сети, объединение ресурсов в пулы, гибкость и контролируемое обслуживание. Пользователи могут получать доступ к ресурсам по мере необходимости, динамически увеличивать или уменьшать объем их использования, регулируя и сохраняя тем самым актуальный для себя баланс стоимости и качества [52].

В то время как ОВ произвели революцию в ИТ-сфере (сфере Информационных Технологий), для некоторых приложений потребовались вычислительные мощности ближе к границе сети. Туманные вычисления (ТВ) [108] стали дополнительной парадигмой облачных вычислений, отвечающей этим требованиям.

ТВ расширяют возможности облака до границ сети, приближая вычисления, хранение и сетевое взаимодействие к источнику данных [21]. Они используют распределенную архитектуру, состоящую из периферийных устройств, шлюзов и облачных ресурсов, образуя континуум от облака до периферийного устройства.

Концепция ТВ позволяет частично преодолеть ограничения традиционных облачных вычислений в сценариях, требующих низкой задержки и быстрого времени отклика, обработки данных в режиме реального времени, эффективного использования полосы пропускания и повышения конфиденциальности и безопасности данных [28]. Благодаря оперативной обработке данных ТВ снижают потребность в масштабной передаче данных на удаленные облач-

ные серверы, что приводит к снижению задержек и ускорению времени отклика, а также эффективному использованию полосы пропускания. Устройства для ТВ (или Fog Computing) также отличаются высокой энергоэффективностью.

Эти функции делают ТВ привлекательными для применения в различных областях, включая Интернет Вещей (IoT – Internet of Things) [153], «Умные города» (УГ) [79], автоматизацию [73; 144], здравоохранение [78; 80; 178] и сельское хозяйство [19; 20]. В таких областях и приложениях, как Интернет Вещей, компьютерная томография и здравоохранение, непрерывное обслуживание имеет решающее значение. Например, в секторе здравоохранения отказ системы медицинского мониторинга может иметь опасные для жизни пациента последствия. Аналогичным образом, в умных городах сбои в работе систем управления дорожным движением могут привести к транспортному коллапсу. Надежность в данном контексте определяется как непрерывность корректного обслуживания, выражающегося, например, в отсутствии ошибок и их маскировки [26]. Для ТВ характерны недостатки, связанные с надежностью [182]. Кроме того, в сфере здравоохранения и Интернета Вещей защита персональных данных имеет первостепенное значение. Термин «безопасность» охватывает способность системы сохранять конфиденциальность данных, что, в свою очередь, означает защиту данных от кражи, копирования и разглашения личности владельца, а также защиту от несанкционированных действий [166]. Подводя итог, можно сказать, что безопасность и надежность имеют фундаментальное значение для функциональности, а, следовательно, и для более широкого внедрения систем ТВ в различные критически важные приложения.

Рассмотрим более подробно облачные технологии (ОТ). По сути, ОТ и ОВ являются синонимами, так как любая обработка данных в облаке предполагает какие-либо вычисления. Аналогично ситуация обстоит с облачными хранилищами, так как при обработке информации (например, поиске) так же производятся определенные вычисления [112]. ОВ являются развитием модели распределенных вычислений [130] за некоторыми исключениями. Распределенные вычисления предполагают наличие параллелизма в вычислениях, а именно объединение вычислительных ресурсов в параллельную вычислительную систему. Реализация такой системы возможна и на одном физическом устройстве, например, серверной стойке или суперкомпьютере [30]. Переходной точкой между распределенными вычислениями и облачными вычислениями можно считать грид вычисления [55]. Главное же отличие ОВ заключается в концептуализа-

ции [53]. В отличие от распределенных и грид вычислений, которые являются прежде всего совокупностью средств и способов решения вычислительно сложных задач, ОВ прежде всего являются сервисом (услугой) по предоставлению возможностей для реализации эластичных вычислений. Принято классифицировать ОВ по видам моделей, в рамках которых предоставляется данная услуга:

- Software as a Service (SaaS) – эта модель подразумевает использование облачной инфраструктуры для реализации ОВ посредством предоставления пользователю пакета прикладного программного обеспечения. Контроль и управление инфраструктурой производится исключительно провайдером услуг [60];
- Platform as a Service (PaaS) – в таком случае пользователю предоставляется инфраструктура для размещения различного базового программного обеспечения. Обычно это инструментальные средства для создания программного обеспечения и, например, системы управления базами данных. Так же провайдером могут предоставляться различные среды для работы с языками программирования [45];
- Infrastructure as a Service (IaaS) – в данном варианте поставщик предоставляет пользователю наиболее полные права по пользованию облаком. Здесь пользователю предлагается базовая инфраструктура, в рамках которой он самостоятельно организует процессы управления вычислительными ресурсами, а также построения сети и хранения данных. Так же пользователь самостоятельно контролирует операционные системы, которые разворачиваются в облаке. Еще одно отличие от предыдущих категорий – пользователю предоставляется ограниченный контроль над сетевыми сервисами выделенного ему облачного пространства [39; 118].

На основе сочетаний трех перечисленных моделей выделяют и различные гибридные, такие как Data Base as a Service [114], Monitoring as a Service [120] и т.п., однако гибридные виды, по сути, разделяются с точки зрения потребностей клиента и скорее направлены на узкую специализацию применения того или иного вида ОВ. На основе вышеизложенной информации можно выделить преимущества, которые дают ОВ для ИИ. Как уже было сказано, методы ИИ являются как правило вычислительно сложными. При проведении исследований и разработке приложений и сервисов, сопряженных с ИИ, исследователь/разработчик сталкивается с трудностями, связанными с ограни-

ченностью вычислительных ресурсов. В случае исследователя, в теории, можно воспользоваться распределенной вычислительной системой своего учреждения, однако не в каждом учреждении имеется собственная мощная вычислительная система, либо к ней затруднен доступ. ОВ позволяют исследователю/разработчику арендовать вычислительные ресурсы у поставщика.

В работе [77] авторы рассмотрели применение ТВ в УГ. Основное внимание в работе было уделено рассмотрению подходов к интеграции ТВ в качестве основной сети УГ. Работа была сосредоточена на сравнении ТВ с другими типами РВС, такими как ОВ и краевые вычисления. Работа организована как обзор других исследований по темам УГ, IoT и ТВ. Обзор носит общий характер, с небольшими уточнениями для платформы, на которой развернуты ТВ. Авторы предложили обратить внимание на редкоупоминаемые параметры, а в качестве решения предложили использовать инструменты платформы. Аналогичная тенденция наблюдалась и в работе [79]. Здесь была рассмотрена интеграция интеллектуальных решений, их взаимосвязь и организация инфраструктуры. Основное внимание было уделено конфигурации устройств. Для обеспечения безопасности, авторы использовали шифры с высоким уровнем защиты, но эти шифры являются сложными в вычислительном отношении. В заключение авторы указали на необходимость проведения дополнительных исследований.

В продолжение обзора ТВ отметим, что основное внимание авторов работы [150] сосредоточено на физической модели и построении системы мониторинга, и большая часть исследований была посвящена вопросам, связанным с физическим воплощением системы. Исследование было основано на уровнях модели OSI (Open System Interconnection), каждому из которых посвящены отдельные исследования. Авторы ограничились ссылками на них. В данном случае был рассмотрен более широкий спектр параметров, включая вопросы безопасности. Например, с точки зрения защиты данных авторы охарактеризовали ТВ как трудно поддающиеся измерению. С точки зрения достоверности, авторы не дали четких формулировок.

Таким образом, основываясь на нескольких обзорах темы ТВ, можно утверждать, что интерес к ТВ значителен, но на данный момент исследователи сосредоточены на построении системы в целом, на том, как подключать устройства, какой протокол передачи данных более эффективен и т.д. Хотя эти

вопросы косвенно относятся к теме данной работы, четкого ответа на вопрос «Как организовать защищенные ТВ?» до сих пор нет.

ТВ – это по-сути парадигма распределенных вычислений, которая расширяет возможности облака до границ сети (рис. 1.1). Хотя ОВ произвели революцию в предоставлении и использовании вычислительных ресурсов, некоторые приложения требуют низкой задержки, обработки данных в режиме реального времени, эффективного использования полосы пропускания и повышения конфиденциальности и безопасности. ТВ приближают вычисления, системы хранения и сетевое взаимодействие к источникам данных для локальной обработки и анализа [21].

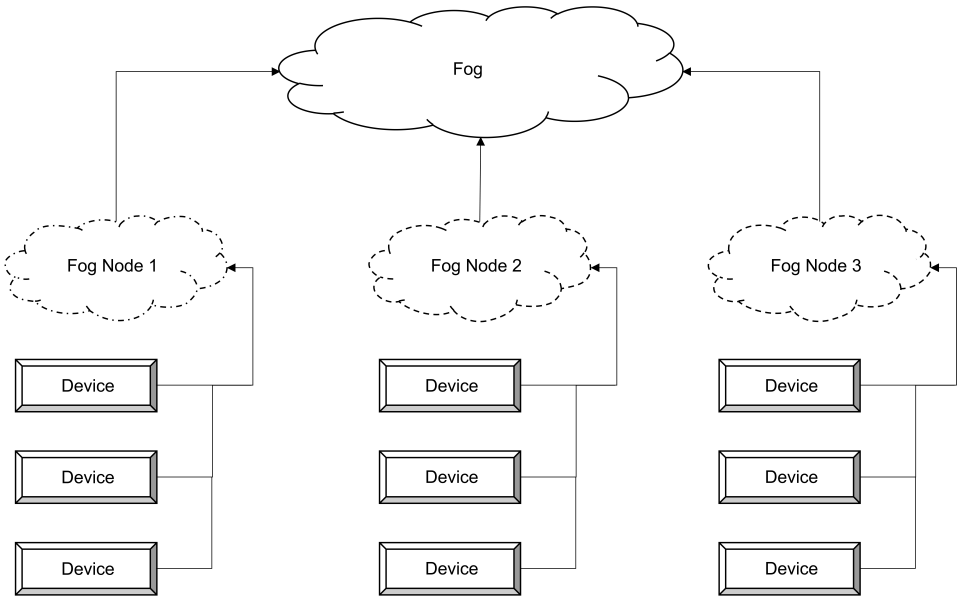


Рисунок 1.1 — Туманные вычисления

В работе [136] исследователи рассмотрели перспективы использования ТВ в приложениях реального времени, и действительно, 10 лет спустя ТВ широко используются в сетях «Умного города» и IoT. В [185] авторы утверждали, что использование ТВ повысит эффективность сетей «Умного города» на базе Интернета Вещей, и предложили многоуровневую архитектуру ТВ. В работе [81] авторы представили дизайн платформы с несколькими приложениями ТВ.

В работе [108] представлено исследование применимости ТВ для медицинских целей, а именно для сопровождения диабета. Авторы сравнили ТВ с ОВ. В своем исследовании они утверждают, что для контроля диабета ТВ более эффективны, чем ОВ, с точки зрения скорости, снижения затрат на создание сети и ее стабильности.

ТВ находят применение в самых разных областях, включая следующие:

- Интернет Вещей: ТВ играют решающую роль в развертывании Интернета Вещей [153]. С увеличением количества подключенных устройств, генерирующих огромные объемы данных, ТВ выводят локальную обработку и аналитику на передний план, сокращая время ожидания и позволяя принимать решения в режиме реального времени. Это облегчает эффективную фильтрацию и агрегацию данных, оптимизирует использование пропускной способности сети и снижает нагрузку на облачные серверы.
- Умные города: ТВ играют важную роль в реализации инициатив УГ [79]. Когда туманные узлы (ТУ) развертываются по всей городской инфраструктуре, данные из различных источников, таких как датчики, камеры и подключенные устройства, могут обрабатываться локально. Это позволяет осуществлять мониторинг, анализ и принятие решений в режиме реального времени для таких приложений, как управление дорожным движением, утилизация отходов и общественная безопасность.
- Автоматизация: в промышленных условиях ТВ обеспечивают обработку и анализ данных в режиме реального времени для критически важных приложений [73; 144]. Периферийные устройства и шлюзы собирают и обрабатывают данные с промышленных датчиков и оборудования, обеспечивая локальное управление и мониторинг. Это сокращает время ожидания, ускоряет время отклика и повышает эффективность в таких отраслях, как производство, энергетика и логистика.
- Здравоохранение: ТВ играют жизненно важную роль в системах здравоохранения, обеспечивая мониторинг, анализ и поддержку принятия решений в режиме реального времени. Современные устройства и шлюзы могут собирать и обрабатывать данные о пациентах, обеспечивая своевременное медицинское вмешательство, удаленный мониторинг пациентов и персонализированные медицинские услуги [78; 80; 178]. ТВ также решают проблемы конфиденциальности, храня конфиденциальные данные локально, обеспечивая соблюдение законодательных норм относительно конфиденциальности личных медицинских данных.
- Сельское хозяйство: в сельскохозяйственном секторе ТВ продвигают точное и интеллектуальное земледелие. Устройства и датчики собирают данные о почве, погоде и состоянии сельскохозяйственных культур, чтобы принимать решения о поливе, внесении удобрений и борьбе с вре-

дителями. ТВ обеспечивают анализ и мониторинг в режиме реального времени для оптимизации использования ресурсов и повышения урожайности [19; 20].

Таким образом, можно сказать, что ТВ и ОВ являются перспективными инструментами, направленными на автоматизацию многих сфер человеческой деятельности. ТВ и ОВ тесно связаны с IoT и УГ как их компоненты.

1.1.2 Искусственный интеллект

Технологии искусственного интеллекта (ИИ) получают все большее распространение в производственной и повседневной жизни общества. Рост популярности и быстрое развитие технологий приводят к усложнению задач, решаемых с помощью ИИ, и, как следствие, к тому, что обработка информации на стандартном пользовательском устройстве выполняется критически неэффективно, что неприемлемо для конечного потребителя. В данном случае выходом является применение облачных технологий (ОТ), которые в свою очередь уже обрели широкую популярность и получили развитие своей методологии. Стоит так же отметить, что ИИ и ОТ вызывают большой интерес и в научном сообществе. Например, такие проекты как [44], [90] обрели популярность в повседневной сфере деятельности, при этом являются научными проектами в области ИИ.

При совместном использовании методов ИИ и ОТ появляется ряд как стандартных, так и специфических проблем, связанных с надежностью, безопасностью и конфиденциальностью. Учитывая, что из всего функционала, реализуемого ОТ, ИИ использует в основном облачные вычисления (ОВ), для ИИ с использованием ОТ имеют место все проблемы, характерные для ОВ. Угрозы безопасности ОВ в целом можно разделить на две категории: внешние угрозы и внутренние. К внешним угрозам относятся различные атаки злоумышленников с целью кражи информации (например, взлом) или с целью повреждения информации (например, DDOS-атаки) [66]. Внутренние угрозы в целом представляют собой совокупность всех возможных способов компрометации системы безопасности изнутри. Для борьбы с последними используют, например, схемы разделения секрета [37]. Однако, наиболее высокий

уровень безопасности достигается при использовании методов, которые позволяют обрабатывать данные в зашифрованном виде. В таком случае вероятность компрометации системы сокращается до минимума. Возможным решением проблемы конфиденциальной обработки данных является гомоморфное шифрование. В работе [38] представлено исследование современных используемых на практике методов и алгоритмов обеспечения конфиденциальности ИИ и предиктивное исследование методов, которые возможно будут применяться в будущем.

ИИ изначально развивался в рамках интеллектуальных систем и до сих пор считается одной из их составляющих, а именно способностью выполнять творческие функции. Изначально необходимость в интеллектуальных системах была обусловлена автоматизацией принятия решений. То есть от системы ожидалась определенная реакция на определенные события. С развитием вычислительной техники, методов и алгоритмов, а также способов разработки систем и приложений данные реакции усложнялись и становились более гибкими. Исторически первыми методами ИИ были методы машинного обучения (МО). МО представляет собой класс методов ИИ, задачей которых является не прямое решение задачи, а нахождение его за счет обучения на основе анализа множества решений схожих задач [117]. Постановку задачи МО можно определить следующим образом. Существует некая неизвестная зависимость между двумя множествами. Известны только прецеденты, т.е. пары из этих двух множеств, которые называются обучающей выборкой. На основе этих данных ставится задача восстановления зависимости, то есть построения алгоритма, который с заданной точностью создаст новую пару. По сути, перед МО ставится задача аппроксимации функции, но не обязательно другой функцией, а неким алгоритмом. Методы ИИ так же можно разделить по способу обучения. Так, например, существуют модели обучения с подкреплением [103], среди таких моделей можно выделить генетические алгоритмы. Генетические алгоритмы представляют собой эвристические алгоритмы поиска, использующиеся для решения задач оптимизации и моделирования, путем случайного подбора, комбинирования и вариации исходных параметров, которые основаны на методах, подобных естественному отбору в природе [169]. Так же существует обучение без учителя, подобными методами решается, например, задача кластеризации [168]. Однако, наибольшую группу методов составляют методы, использующие обучение с учителем, где для множества прецедентов (известных пар входных

и выходных данных) необходимо построить алгоритм, возвращающий требуемое решение [59].

Искусственные нейронные сети или просто нейронные сети (НС) – представляют собой математическую модель, которая построена по принципу функционирования сетей нервных клеток живого организма [180]. Развитие вычислительной техники позволило создавать модели НС большой сложности. В этом контексте можно выделить несколько событий, которые позволили расширить применение ИИ, это появление различных аппаратных и графических ускорителей [34; 123; 135], а так же вентильных матриц [48; 119; 149; 158] для ИИ, которые позволяют решать более широкий спектр задач, недоступный ранее. Работа таких моделей НС осуществляется за счет применения, так называемого глубокого обучения (ГО). По сути, ГО это процесс обучения многослойных НС. Теоретически 2-3-слойных НС достаточно для решения широкого круга задач, однако, для решения сложных задач зачастую используется ГО, которое показывает хорошие результаты [88]. К ГО относят такие методы как ограниченная машина Больцмана. Так же на основе ГО строятся сверточные нейронные сети, которые используют на различных слоях различные формы сверток [42]. Применяются они зачастую для распознавания образов [145].

Одной из наиболее популярных современных технологий ИИ является технология GPT (Generative Pre-trained Transformer). Даже в начале развития GPT модели требовали больших обучающих выборок, которые занимают десятки гигабайт на стадии предварительного обучения, а учитывая тот факт, что количество пользователей моделей с GTP превышает 100 миллионов, объемы потребляемых ресурсов превышают ресурсы доступные одному устройству, если рассматривать среднестатистический офисный компьютер. Для эффективной работы таких технологий требуется применение распределенных вычислительных систем. Рассмотрим НС подробнее.

Искусственная нейронная сеть (НС) – математическая модель биологической нейронной сети. Представляет собой совокупность связанных между собой искусственных нейронов [170]. Каждый искусственный нейрон (далее «нейрон») представляет собой по-сути имитационную модель биологического нейрона, который возбуждается при получении определенного заряда, в качестве аксона (передатчика) выступают функции активации, которые передают значения дальше при достижении определенного порогового значения. Каждый нейрон обладает весом w, «внутренним зарядом», получаемая от других

нейронов информация x изменяет значение w в зависимости от типа функции активации, как например ReLU (Rectified Linear Unit) [69]: если $w \cdot x > 0$, то далее передается $w \cdot x$, иначе передается 0. Тогда НС можно описать простой формулой:

$$nn = \sum_{i=1}^{n} w_i \cdot x_i. \tag{1.1}$$

Нейроны принято группировать по слоям. Если каждый нейрон текущего слоя связан с каждым нейроном предыдущего слоя и каждым нейроном последующего слоя – такая НС называется полносвязной, и такие НС наиболее распространенные. Помимо функции активации ReLU существует, как несколько модификаций ReLU, так и другие, например, сигмоидальная $\frac{1}{1+e^{-ax}}$, гиперболический тангенс $\tanh \frac{x}{a}$ и многие другие [69]. В зависимости от задачи, которую решает НС для нее будет более эффективна та или иная функция активации.

Главной особенностью НС, как и большинства методов искусственного интеллекта, является необходимость ее обучения. Наиболее популярный алгоритм обучения – это метод обратного распространения ошибки [54]. Метод исходит из того, что возможно оценить качество работы НС, тогда обучение сети можно свести к задаче оптимизации. Для оценки работы нейронной сети (1.1) применяются различные подходы, но наиболее распространенным является средний квадрат разности между выходными значениями и требуемыми

$$E(w_{i,j}) = \frac{1}{2} \sum (t - outputs)^2, \tag{1.2}$$

после чего для модификации весов сети производится градиентный спуск

$$\Delta w_{i,j} = -\eta \frac{\partial E}{\partial w_{i,j}}.$$

Первыми модифицируются нейроны последнего слоя, благодаря чему метод и получил свое название. Данный метод хоть и обладает некоторыми недостатками, такими как возможный уход в локальный минимум и паралич сети, при правильном использовании и выборе шага η показывает хорошие результаты.

За более чем пол века было разработано большое количество видов и моделей НС. Для настоящего исследования наибольший интерес представляют сверточные НС (СНС) [23]. Основная идея СНС заключается в подражании зрительной коре головного мозга. Свое название модель получила из-за

многократно выполняемой операции свертки и более всего подходит для распознавания образов.

Свертку можно описать следующим образом. Изображение представляется матрицей $M \times N$, где M, N – ширина и высота изображения в пикселях, т.е. каждому пикселю соответствует элемент матрицы. Если изображение черно-белое, т.е. представлено в градациях серого, то оно называется одноканальным и матрица одна, если изображение цветное, т.е. представлено в формате RGB, тогда оно называется трехканальным и общий размер матриц определяется как $3(M \times N)$. При высоком разрешении изображения обычному персептрону потребовалось бы огромное количество входных нейронов. Свертка позволяет уменьшить размер матрицы с сохранением признаков изображения. Например, если имеется трехканальное изображение 100×100, то с ядром свертки 3×3 и 1 канальным выходом количество входных нейронов уменьшится с 30000 до 9604. Количество входных нейронов так же можно сократить за счет пулинга (pooling) [18]. Если используется Максимальный пулинг (MaxPooling), тогда матрица $M \times N$ разбивается на подматрицы $m \times n$, где M кратно m, а N кратно n, после чего из соседних подматриц, выбираются максимальные значения, из которых собирается матрица меньшего размера. В пулинге по среднему – отличием является то, что вместо максимального элемента подматрицы находится среднее значение элементов подматрицы.

СНС обладает весомым преимуществом над типовой моделью НС (персептроном) в том смысле, что за счет операции свертки на одно изображение требуется меньшее количество весов w (т.е. нейронов) чем было бы необходимо для персептрона, однако, стоит отметить, что и для типовых моделей НС проводятся исследования по повышению их эффективности [128; 131]. СНС эффективно распараллеливается, кроме того, за счет свертки имеет устойчивость к поворотам, сдвигам и прочим нестабильностям изображений. Из минусов СНС можно выделить сложность настройки: если веса обучаются, то гиперпараметры сверточных слоев определяются разработчиком, при этом необходимо учитывать такие операции как пулинг, слои субдискретизации и т.п. Данные гиперпараметры существенно влияют на точность работы сети.

Рассмотрим применимость различных моделей ОВ для ИИ. Основные результаты исследований данного вопроса представлены в работе [31], рассмотрим результаты этого исследования тезисно. В целом ИИ может быть развернут в рамках любой из трех моделей предоставления услуг. Однако есть несколько

нюансов, в случае SaaS поставщик помимо вычислительных ресурсов так же поставляет и ИИ, в данном случае ИИ так же является услугой. Примером такой услуги является MLaaS (Machine Learning as a Service) [126]. В случае PaaS поставщик предоставляет пользователю, помимо вычислительных мощностей, инструментарий по разработке ИИ. В данную категорию можно отнести различные облачные сервисы для разработки программного обеспечения (ПО). Ярким примером является Google Colab [40]. Несмотря на то, что данное решение не является специализированным, оно все же предоставляет пользователю возможности по разработке и исследованию ИИ. При этом существуют решения, направленные именно на работу с ИИ [91]. IaaS и ИИ достаточно трудно выделить в специализированную категорию, так как в данном случае поставщик услуг предоставляет исключительно вычислительные ресурсы и инфраструктуру связывающую их. Здесь специализация на ИИ достигается за счет двух факторов: за счет того, что аппаратное обеспечение технической составляющей ОВ подобрано таким образом, чтобы ИИ работал максимально эффективно; а также за счет самой позиции поставщика услуг [132]. В рассмотренных случаях ИИ выступает как объект, либо являющийся частью услуги, либо являющийся фактическим потребителем вычислительных мощностей, однако существуют работы, в которых рассматриваются и другие возможности по применению ИИ в ОВ [124; 125; 173; 181]. Например, ИИ может применяться для построения инфраструктуры и балансировки нагрузки.

Рассмотрим удаленную обработку данных СНС. Удаленная обработка подразумевает передачу пользователем данных удаленному сервису. Однако, данные, которые представляют определенную ценность для пользователя, либо защищаемые законом (например, личные или медицинские данные третьих лиц), объединяемые понятием конфиденциальные данные, передавать в открытом виде для обработки удаленным сервисам нельзя. С другой стороны, удаленный сервис не может допустить передачу параметров обученной CNN пользователю для локальной обработки данных на его машине, поскольку настройка данных параметров (коэффициентов фильтров на сверточных слоях, коэффициентов реализующих классификацию слоев модели CNN и т.д.), учитывая сложность современных CNN, требует больших финансовых вложений.

Сервисы, предоставляющие удаленный доступ к обученной СНС, функционируют по условному принципу «загрузили данные, выгрузили результат». Основным приложением СНС является распознавание образов (классификация

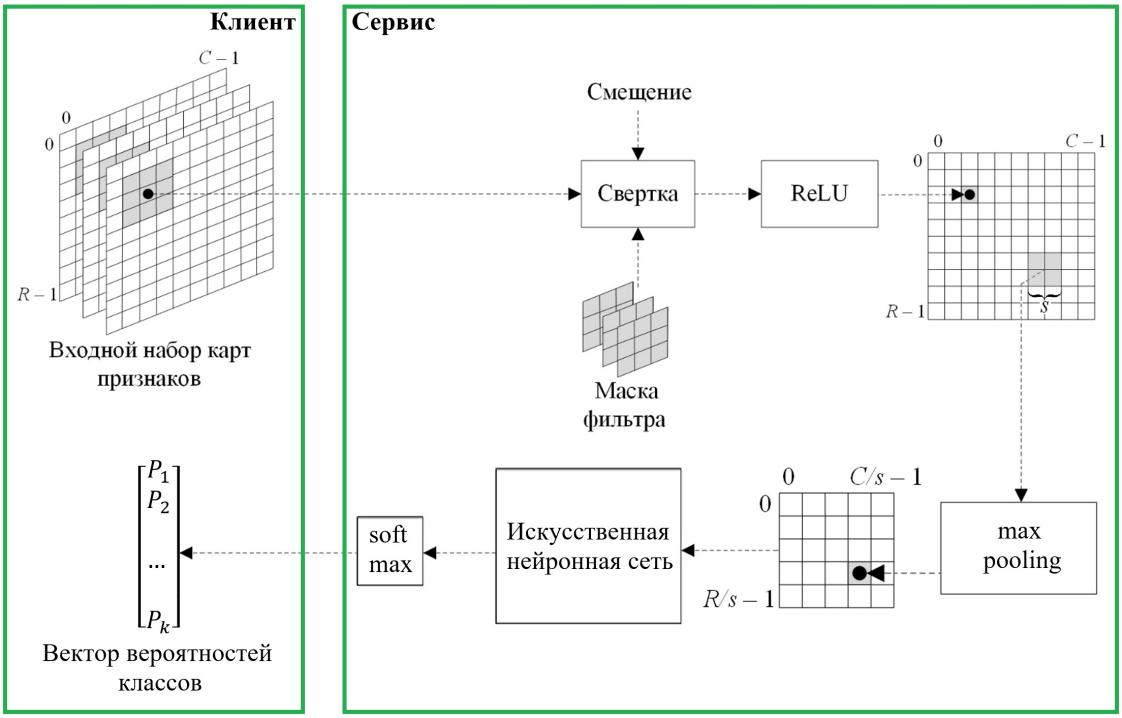


Рисунок 1.2 — Упрощенная архитектура CHC для удаленной обработки открытых данных

изображений), поэтому на вход подаются визуальные данные, а на выходе получается вектор вероятностей принадлежности к каждому из классов (к каждой из групп схожих в каком-то смысле изображений). Под визуальными данными понимается массив значений интенсивностей базисных цветов, задающий цвет каждого из пикселей в соответствии с используемой цветовой схемой.

На рисунке 1.2 отделены клиентская и сервисная части процесса удаленной обработки данных с помощью CHC. Клиент должен лишь загрузить данные и выгрузить результат.

Отметим, что представленная схема является упрощенной в сервисной части, поскольку как правило используется множество фильтров и слоев max pooling, а также множество слоев свертки в разном сочетании в зависимости от конфигурации CHC (обобщение тривиально).

1.2 Проблема безопасности облачных и туманных вычислений

Ключевым аспектом безопасности в этом исследовании является способность системы сохранять конфиденциальность данных, что, в свою очередь, означает, что данные защищены от кражи, копирования, разглашения личности владельца и несанкционированных действий [166]. Безопасность ТВ и ОВ, в целом, основана на безопасности РВС, поскольку ОВ и ТВ являются развитием этой модели вычислений. Важно обеспечить конфиденциальность данных не только снаружи системы, но и внутри между соседними узлами. Существует ненулевая вероятность неблагонадежности «владельцев» вычислительных узлов, поэтому злоумышленник может получить контроль над одним или несколькими узлами. Кроме того, ТВ – это сеть узлов с низким энергопотреблением, что является фактором, ограничивающим вычислительную сложность методов обеспечения безопасности. Таким образом, вопрос безопасности ОВ и ТВ является актуальным предметом исследований в научном сообществе [147].

Если проекты, как в случаях [44; 90], являются открытыми, т.е. изначально не содержат конфиденциальной информации, то требования к их безопасности невелики. Однако, любой ИИ может применяться в задачах, связанных с обработкой конфиденциальных данных. Это может быть медицинская информация [47; 109; 110; 115; 154; 179], банковские [27; 33; 151], государственные [22; 146] и всевозможные личные данные. В данных случаях ОВ часто подвергаются критике как в обществе в целом, так и в научном сообществе. При обработке какой-либо информации в облаке к ней имеют доступ обе стороны – пользователь и поставщик услуги. В этом случае степень конфиденциальности данных устанавливается соглашением между пользователем и поставщиком, а то, как соблюдается данное соглашение находится в зоне ответственности поставщика услуг. Не редки случаи, когда конфиденциальные данные, хранимые или обрабатываемые различными поставщиками услуг, были скомпрометированы [161]. В таком случае пользователь вынужденно отдает предпочтение поставщикам, которые либо зарекомендовали себя, как надежные с точки зрения обеспечения конфиденциальности обрабатываемых данных, либо утверждают о низкой вероятности компрометации за счет применения эффективных методов защиты [25]. Для более наглядной демонстрации ретроспективного обзора распределенных вычислительных технологий и ИИ была

составлена Таблица 1. Рассмотренные выше работы представлены не в хронологическом порядке, Таблица 1 призвана его восстановить. Отметим, что некоторые технологии впервые представлены намного раньше, чем они получили развитие и были изучены в полном объеме [31].

Таблица 1 — Историческая справка по рассмотренным технологиям РВС, ОВ и ИИ [31]

Год	Распределенные вычисления	Облачные технологии	Искусственный интеллект
1943	–	–	Концепция НС [105; 137]
1954	–	–	Зарождение генетических алгоритмов [156]
1959	–	–	Машинное обучение [152]
1966	–	–	Появление языковых моделей [86]
1978	Принципы распределения работы между процессорами [2]	–	–
1980	–	–	Теоретическое описание ГО [98]
1992	Зарождение GRID [164]	–	–
1996	Проект GIMPS по поиску целых чисел [46]	–	–
1999	Проект SETI на базе BOINC [32]	–	–
2000	–	–	Начало практического применения ГО [63], Компьютерное зрение [62]
2006	–	Зарождение концепции облачных вычислений [52]	–
2008	–	Определение концепции облачных вычислений как услуги [76]	–
2009	–	Запуск Google Apps [95]	–
2011	–	Стандартизация SaaS, PaaS и IaaS как моделей обслуживания в ОВ [39; 45; 60]	–

Продолжение таблицы 1

Год	Распределенные вычисления	Облачные технологии	Искусственный интеллект
2015	–	Развитие туманных вычислений, как основы для Интернета Вещей [65], Запуск OpenFog [107]	–
2018	–	–	GPT [143]

Анализируя данные Таблицы 1 и представленный выше обзор, можно заметить, что теоретические модели ИИ активно развивались с середины XX века, однако практические модели получили свое развитие в конце XX – начале XXI веков. Так же из таблицы видно, что развитие практических моделей ИИ совпадает по времени с поздними этапами развития распределенных вычислений (до конца XX века) и зарождением ОВ в начале XXI века. Данное совпадение не случайно и связано с тем, что развитие распределенных и облачных вычислений все это время шло по пути агрегирования все больших вычислительных мощностей, которых так не хватало большим моделям ИИ. Таким образом справедливо утверждение, что распределенные вычисления внесли немалый вклад в развитие моделей ИИ, а ОВ позволяют развивать и создавать современные вычислительно затратные модели ИИ, сложность которых растет с каждым днем. Учитывая вышесказанное, можно утверждать, что проблема конфиденциальности ОВ имеет такое же значение для ИИ, как для любого «потребителя» ОВ, коим ИИ по сути и является. Далее будут рассмотрены методы обеспечения конфиденциальности ИИ в ОВ.

Анализ публикационной активности по вопросам безопасности (табл. 2) позволяет констатировать большой интерес к данной тематике. Некоторые публикации цитировались более тысячи раз [147].

Таблица 2 — Публикации связанные с безопасностью ТВ и ОВ [147]

Название	Цитируемость	Загрузок	Издатель
Fog Computing Security: A Review of Current Applications and Security Solutions	440	49000	Springer
An Overview of Fog Computing and Its Security Issues	484	-	Wiley
The Fog Computing Paradigm: Scenarios and Security Issues	1369	13222	IEEE

Продолжение таблицы 2

Название	Цитируемость	Загрузок	Издатель
Security and Privacy Issues of Fog Computing: A survey	661	7687	Springer
Security and Trust Issues in Fog Computing: A survey	305	335	Elsevier
Centralized Fog Computing Security Platform for IoT and Cloud in Healthcare System	236	-	IGI Global
A Fully Homomorphic Encryption Scheme	3989	-	Stanford
Homomorphic Encryption for Arithmetic of Approximate Numbers	1809	27000	Springer
Ensemble Method for Privacy-Preserving Logistic Regression Based on Homomorphic Encryption	78	2909	IEEE
Efficient Homomorphic Comparison Methods with Optimal Complexity	116	2914	Springer
FPGA-Based Accelerators of Fully Pipelined Modular Multipliers for Homomorphic Encryption	43	1791	IEEE
Implementation and Performance Evaluation of RNS Variants of the BFV Homomorphic Encryption Scheme	116	1445	IEEE
A Homomorphic Encryption Scheme for Cloud Computing Using Residue Number System	93	879	IEEE
High-Precision Bootstrapping of RNS-CKKS Homomorphic Encryption Using Optimal Minimax Polynomial Approximation and Inverse Sine Function	76	3837	Springer
Secret-Sharing Schemes: A Survey	826	3643	Springer
How to Share a Secret	19438	29420	ACM
A Modular Approach to Key Safeguarding	975	1802	IEEE
How to Share a Secret	455	1710	Springer

По результатам анализа данных работ, а так же работы [104], можно выделить наиболее часто освещаемые угрозы безопасности ОВ и ТВ:

1. Продвинутые постоянные угрозы (APT – Advance Persistent Threats) – это кибератаки, направленные на компрометацию инфраструктуры компании с целью кражи данных и интеллектуальной собственности.
2. Проблемы с контролем доступа (ACI – Access Control Issues) могут привести к плохому управлению, и любой неавторизованный пользователь

сможет получить данные и разрешения для установки программного обеспечения и изменения конфигураций.

3. Перехват учетной записи (AH – Account Hijacking) – это атака, направленная на захват учетных записей пользователей в злонамеренных целях. Фишинг – это потенциальный метод захвата учетной записи.
4. Отказ в обслуживании (DoS – Denial of Service) – это когда законным пользователям не разрешается пользоваться системой (данными и приложениями) из-за чрезмерного использования ограниченных системных ресурсов.
5. Утечка данных (DB – Data Breaches) – это когда злоумышленник разглашает или крадет важные, защищенные или конфиденциальные данные.
6. Потеря данных (DL – Data Loss) – это случайное (или злонамеренное) удаление данных из системы. Это не обязательно должно быть результатом кибератаки и может быть вызвано стихийным бедствием.
7. Небезопасные API (IA – Insecure APIs). Многие поставщики облачных услуг предоставляют клиентам интерфейсы прикладного программирования (API). Безопасность этих API имеет решающее значение для безопасности любых реализованных приложений.
8. Системные и прикладные уязвимости (SAV – System and Application Vulnerabilities) – это уязвимости, возникающие в результате ошибок конфигурации в рекламном программном обеспечении, которые злоумышленник может использовать для проникновения и компрометации системы.
9. Злоумышленник-инсайдер (MI – Malicious Insider) – это пользователь, который получил авторизованный доступ к сети и системе, но осознанно решает действовать злонамеренно.
10. Недостаточная комплексная проверка (IDD – Insufficient Due Diligence) часто возникает, когда организация спешит спроектировать и внедрить систему.
11. Злоупотребление и недобросовестное использование (ANU – Abuse and Nefarious Use) часто происходит, когда ресурсы предоставляются бесплатно, и злоумышленники используют эти ресурсы для осуществления вредоносных действий.

12. Проблемы совместимости (STI – Shared Technology Issues) возникают из-за совместного использования инфраструктур, платформ или приложений. Например, базовые аппаратные компоненты, возможно, не были спроектированы таким образом, чтобы обеспечивать высокие изоляционные свойства.

В работе [104] представлен анализ влияния рассматриваемых угроз на различные приложения на базе ТВ.

Аналогичные исследования приведены в работах [29; 49; 171; 183; 186]. В целом эти исследования можно охарактеризовать следующим образом. Исследователи исходили из возможных областей применения ОВ и ТВ, и сосредоточились на их особенностях. С одной стороны, это правильное направление, поскольку, если сравнивать ТВ для Интернета Вещей и ТВ для здравоохранения, это две совершенно разные области, требующие разных подходов. Однако сама основа системы у них одинакова и методы обеспечения базового уровня безопасности идентичны. Проанализировав угрозы, можно увидеть, что, например, угроза DB для большинства приложений такая же, как и угроза ACI. Таким образом, можно сказать, что основным требованием к безопасности данных является их конфиденциальность. Даже если злоумышленнику удастся завладеть той или иной частью данных, эти данные не должны иметь для него никакой ценности [147].

1.3 Методы обеспечения безопасности облачных и туманных вычислений

1.3.1 Дифференциальная конфиденциальность

Понятие конфиденциальности в некоторой степени является не строгим. В зависимости от ситуации она может рассматриваться с разных сторон. Например, с точки зрения медицины достаточно, чтобы данные были анонимны, т.е. история болезни была обезличена. В работе [140] проводится исследование методов ГО для обеспечения мягкой конфиденциальности. Авторы достигают своей цели путем внедрения дифференциальной конфиденциальности, т.е.

путем перемешивания частных обезличенных данных вместе с синтезированными данными. В работе продемонстрирована высокая точность данного метода. Однако метод имеет ряд недостатков: основой конфиденциальности является внесение дополнительного шума и увеличение расстояний между точками во время выполнения градиентного спуска. Это не только повышает вычислительную сложность расчетов, но и не обеспечивает надлежащей безопасности данных, так как если злоумышленник получит доступ к данным во время обработки – он сможет убрать лишний шум.

С другой стороны, в работе [162] рассмотрен принципиально отличающийся от вышеприведенного подход к обеспечению конфиденциальности данных. Авторы рассматривают возможность обучения и использования нейронной сети группой пользователей без необходимости раскрытия конфиденциальных данных друг другу. Подобная возможность обеспечивается особенностью стохастического градиентного спуска, которая позволяет выполнять его параллельно и асинхронно. Авторы утверждают, что их решение позволяет участникам обучать нейронную сеть независимо на своих собственных (конфиденциальных) наборах, делясь подмножествами ключевых параметров. В целом, данный метод позволяет обеспечить конфиденциальность среди участников группы, однако, оказывается неэффективным в случае предварительного сговора участников либо внешней атаки.

1.3.2 Схемы разделения секрета

В контексте построения конфиденциального ИИ в ОВ необходимо так же рассмотреть схемы разделения секрета (СРС) [37]. Приведем кратко основные теоретические аспекты, связанные со СРС. Секрет S разделяется дилером D между N участниками таким образом, что для дешифровки какой-либо информации потребуется объединение долей S_i всех участников схемы обратно в секрет, $i \in 2, \ldots, N$, где N – количество участников. Если секретом является, например, ключ для схемы шифрования, конфиденциальность системы возрастает.

Существует два типа схем разделения секрета: полные [37] и пороговые [157]. Полные предполагают, что для восстановления необходимы все доли сек-

рета, пороговые же, что необходимо определенное количество долей, но не обязательно все. Пороговые схемы разделения секрета на практике используются гораздо чаще ввиду своей гибкости. Пороговые СРС позволяют снизить вычислительную нагрузку, при этом порог задается таким, чтобы злоумышленник не смог овладеть необходимым количеством долей секрета, либо чтобы не смог состояться предварительный сговор необходимого для восстановления секрета количества участников. В работе [68] авторы предлагают распределенное ГО, когда участники обучают нейронную сеть с помощью своих конфиденциальных наборов. Авторы приводят результаты исследования, в котором удалось сохранить конфиденциальность при распределенном ГО в облаке с недоверенными участниками. Работа демонстрирует функционал, предоставляемый СРС при работе с ИИ, и сам факт возможности конфиденциального обучения на примере схемы Shamir [157]. За время своего существования схема доказала свою пригодность с точки зрения безопасности. Однако, если рассматривать применение СРС с точки зрения ОВ, то на проблемы безопасности также накладываются проблемы надежности, дополнительные корректирующие коды ведут к дополнительным нагрузкам на систему. Для нивелирования необходимости применения дополнительных корректирующих кодов можно применять СРС, основанные на системе остаточных классов (СОК) [9], например, такие как СРС Asmuth-Bloom [35] и СРС Mignotte [122].

В работах [165] и [67] представлено исследование системы, основанной на СРС для УГ. В первой работе представлены основные концепции и методика построения системы безопасности УГ на базе СРС, в следующей работе полученные результаты были развиты до разработки протокола передачи данных, который базируется на беспроводных децентрализованных самоорганизующихся сетях. Однако, стоит отметить, что основной акцент сделан на повышении надежности системы УГ при сохранении требуемого уровня безопасности, а не его повышении.

В работе [148] рассмотрена организация федеративного обучения нейронных сетей в облачных системах с применением СРС Asmuth-Bloom для обеспечения конфиденциальности [36]. Так же авторами рассматривается СРС Mignotte, однако существуют работы, доказывающие ее непригодность для обеспечения безопасности. В работе [138] авторы предлагают протокол конфиденциальной передачи данных в условиях работы с нейронными сетями. Протокол передачи основан на СРС. Основной упор в статье сделан на ско-

рость обработки данных при сохранении их конфиденциальности. Авторами показана эффективность их решения по сравнению с некоторыми методами гомоморфного шифрования, которое так же применимо при построении конфиденциального ИИ в ОВ.

1.3.3 Полностью гомоморфное шифрование

Гомоморфное шифрование известно с середины XX века. Такие схемы, как RSA [41] и ElGamal [71] обладали гомоморфностью либо по сложению, либо по умножению. Теоретическая возможность создания схемы ПГШ была известна еще тогда, но только в 2009 году Gentry в своей диссертации представил схему ПГШ [84]. Изначально схемы ПГШ были основаны на криптографических решетках [84] и имели очень слабую производительность. Предложенная Gentry схема имела ценность именно в том, что она доказала саму возможность реализации ПГШ. Первые схемы Gentry, по сути, представляли собой криптографический аппарат, способный выполнять гомоморфные операции над полем булевых значений в логических функциях. Применение других методов криптографии, таких как обучение с ошибкой и обучение с ошибкой в кольце (OOK) [50], позволило повысить эффективность будущих схем ПГШ [85]. Позднее последователи Gentry разработали целочисленное ПГШ, примерами являются схемы BGV [43] и BFV [75], а также ПГШ для рациональных чисел с фиксированной точностью, а именно схему CKKS [13]. Некоторые из этих схем были ускорены благодаря использованию COK [101].

Отметим, что деление на ГШ и ПГШ не совсем корректно с точки зрения терминологии. ПГШ поддерживает произвольное количество гомоморфных операций сложения и умножения, однако это не эффективно с точки зрения вычислительной сложности. Компромиссом между ПГШ и обычным ГШ является частично гомоморфное шифрование (ЧГШ) [24]. ЧГШ предполагает выполнение одной гомоморфной операций (зачастую сложения) произвольное количество раз для одного ключа шифрования и определенное количество выполнений другой гомоморфной операции (умножения). С теоретической точки зрения это хороший компромисс. Однако с практической точки зрения несколько десятков разрешенных умножений – очень малое количество, если

рассматривать НС – ничтожно малое. Для решения данной проблемы была предложена операция бутстраппинг (bootstrapping) [82], которая стирает грань межу ЧГШ и ПГШ. Бутстраппинг позволяет расширить количество разрешенных операций, за счет пересчета ключа. Однако, данная операция является вычислительно сложной, поэтому злоупотреблять ею нельзя. Бутстраппинг присутствует во всех популярных современных схемах ПГШ, в том числе в BFV и CKKS. Для эффективной реализации конфиденциальных вычислений нужно учитывать зависимость между предполагаемым количеством операций бутстраппинга и первоначальными параметрами ключа, которые определяют мультипликативную глубину. При выборе параметров для построения безопасной и эффективной схемы ПГШ исследователи обычно обращаются к стандарту ПГШ, разработанному сообществом ПГШ [87].

Набор из операций сложения и умножения позволяет реализовать большое количество алгоритмов ИИ, что было достаточно быстро обнаружено многими исследователями ИИ. В работе [10] авторы проводят исследование применимости схем ПГШ в НС. Обзор демонстрирует как перспективы применения схем ПГШ в НС, так и вызовы, которые являются основным нарративом в данной работе. Если тезисно их выделить, то можно заключить следующее. Основным препятствием является вычислительная сложность методов ПГШ, что значительно снижает производительность зашифрованной НС. Ограничение на операции требует изменения подхода к проектированию НС, а именно к реализации функций активации; выходом из этой ситуации является применение приближенных методов, например, построение полиномиальных функций активаций, так как для вычисления полинома требуются только операции сложения и умножения. В целом, применяя приближенные функции активации и оптимизируя модель нейронной сети, можно добиться достаточного уровня производительности при необходимой точности. Рассмотрим несколько работ, в которых авторы демонтируют практические результаты исследований, связанных с ПГШ и НС.

CryptoNets [57]. В данном решении авторы предлагают применение ГШ в работе конфиденциального облачного хранилища с возможностью обработки нейронными сетями. Основным достигнутым результатом является точность нейронной сети по распознаванию образов, равная 99%.

В случае SEALion [72] предлагается решение, основанное на TensorFlow [175] и SEAL [121]. TensorFlow реализует тензорные вычисления, SEAL само

ГШ. Разработанное решение авторы используют для СНС, в качестве метода обучения применяется метод опорных векторов. В работе представлены несколько моделей НС, точность определяется на основе качества распознавания изображений цифр. Авторы показывают, что их решения более эффективны чем, например, в работе [57]. Похожим решением является TenSEAL [174], эта библиотека предоставляет возможности по работе с МО, НС и СНС совместно с ГШ. В качестве недостатка можно выделить сложности при обучении на зашифрованных данных, при этом обученная НС демонстрирует быстродействие и дает точные результаты.

В работе [139] рассматриваются вариации конфиденциальных СНС с ГШ. Авторами предлагается СНС, в которой применяется целочисленная схема ПГШ – BGV [43]. Основной акцент сделан на возможности вычисления значения функции активации ReLU от шифртекстов, полученных согласно схеме BGV. Авторами было определенно три требования для НС – точность, конфиденциальность и эффективность, результатом работы является полиноминально приближенная функция ReLU, которая позволила уменьшить мультипликативную глубину, тем самым повысив эффективность работы сети. Данную работу можно считать достаточно важной по нескольким причинам: во-первых, показан сам факт возможности полностью конфиденциальных расчетов; во-вторых, показана возможность построения и применения нейронных сетей с ПГШ.

В работе [141] рассматривается сохраняющее конфиденциальность ГО. Авторы противопоставляют свои исследования многим работам, которые были рассмотрены ранее, сосредотачиваясь на устранении недостатков, которые были озвучены выше. Основной результат базируется на применении методов аппроксимации для различных функций активации, например, таких как ReLU и Softmax. Для обеспечения большого количества гомоморфных умножений в схеме CKKS авторы предлагают свою модификацию процедуры бутстраппинг, которая основана на том факте, что схема CKKS [13] применяет систему остаточных классов для ускорения арифметических вычислений. Этот результат интересен тем, что повышает быстродействие методов ГШ для конкретных задач, тогда как повышение быстродействия и применимости ГШ в целом является основным направлением исследований в этой области в настоящее время.

ГШ позволяет решить проблемы безопасности ОВ и ОТ в целом, создав прозрачную среду для свободных и конфиденциальных вычислений в облаках.

Проведенный анализ методов ГШ в ИИ, позволяет утверждать, что такого мнения придерживаются многие исследователи. Выше рассмотрены схемы ПГШ, которые в большей степени основаны на криптографических решетках, однако существуют схемы ПГШ, которые построены на СРС. Такой подход позволяет расширить возможности ПГШ в ОТ. Далее рассмотрены гибридные модели, которые совмещают методы ПГШ и СРС.

1.3.4 Сравнительный анализ методов обеспечения конфиденциальности нейронных сетей

В ходе исследования были проанализированы три группы методов обеспечения конфиденциальности искусственного интеллекта (табл. 3). Данные в Таблице 3 основаны на информации, представленной в обозреваемых работах [31].

Таблица 3 — Результаты аналитического обзора методов обеспечения конфиденциальности ИИ [31]

Метод	Вычислительная сложность	Конфиденциальность при передаче данных	Конфиденциальность при обработке данных	Обеспечение надежности	Рассмотренный метод ИИ
Модифицированный градиентный спуск	$O(D^2N + (D + \varepsilon)^3)$, где D – количество участников, ε – максимальный сдвиг набора данных, который может наблюдаться при добавлении или удалении одного участника	низкая	средняя	Отсутствует	AC-GAN [140]
Асинхронный градиентный спуск	$O(D^2 \cdot N \cdot \theta + D^3)$, где θ – отношение общего числа параметров СНС к скорости выбора параметров СНС	низкая	средняя	Отсутствует	СНС [162]
СРС Shamir	$O(N^2)$	высокая	низкая	Отсутствует	ГО [68]
СРС Asmuth-Bloom	$O(\log_2^2 N + \log_2 N^2)$	высокая	низкая	Корректирующие коды СОК	Федеративное обучение [138; 148]
СРС Mignotte	$O(N^2)$	низкая	низкая	Корректирующие коды СОК	Федеративное обучение [148]

Продолжение таблицы 3

Метод	Вычислительная сложность	Конфиденциальность при передаче данных	Конфиденциальность при обработке данных	Обеспечение надежности	Рассмотренный метод ИИ
BFV	$O(N \log N)$	высокая	высокая	Корректирующие коды СОК (в теории)	НС, СНС [57; 72]
BGV	$O(N \log N)$	высокая	высокая	Отсутствует	НС, СНС [139]
CKKS	$O(N \log N)$	высокая	высокая	Корректирующие коды СОК (в теории)	НС, СНС, ГО [57; 72; 141; 174]

Анализируя результаты можно заметить, что наилучшая конфиденциальность достигается при использовании схем ПГШ, так как оно позволяет обрабатывать данные в зашифрованном виде. Наибольший интерес научного сообщества вызывает схема CKKS, так как она позволяет обрабатывать данные, представленные в виде вещественных чисел. Однако, в данной схеме необходимо уделять внимание точности представления чисел, так как если точность будет обеспечиваться на недостаточном уровне, то вычисления будут проводиться с ошибкой. Для дальнейших исследований необходимо более подробно рассмотреть схемы ПГШ.

1.4 Схемы полностью гомоморфного шифрования

Современное ПГШ имеет множество реализаций в виде криптографических схем. В целом схемы можно разделить на три группы:

1. Схемы, использующие булеву алгебру [84; 85; 177].
2. Схемы, использующие целочисленную арифметику [43; 75].
3. Схема, использующая вещественную арифметику [13].

Однако, в настоящий момент, популярность в научном сообществе получили лишь следующие схемы:

– Схема Fast Fully Homomorphic Encryption Over the Torus (TFHE) – использующая булеву алгебру;
– Схема Brakerski–Fan–Vercauteren (BFV) – использующая целочисленную арифметику;
– Cheon–Kim–Kim–Song (CKKS) – использующая вещественную арифметику с фиксированной запятой.

Популярность в научном сообществе оказала высокое влияние на данные схемы: проводится активное их исследование, предлагаются различные реализации. Учитывая данный факт, было решено использовать в работе именно эти схемы, поскольку их реализации в виде библиотек программного кода позволяют исследовать и применять их для НССК (Нейронных Сетей Сохраняющих Конфиденциальность). Рассмотрим перечисленные схемы.

1.4.1 Схема TFHE использующая булеву алгебру

Главной отличительной особенностью схемы TFHE [177] является то, что для организации своей арифметики она использует так называемые таблицы поиска (LUT – Look Up Table). Их применение позволяет избежать трудностей, возникающих при работе с другими схемами, такими как BFV и CKKS, которые хоть и поддерживают гомоморфное сложение и умножение, но имеют ограниченное количество разрешенных гомоморфных умножений. LUT позволяет реализовать, как гомоморфное умножение, так и гомоморфное сложение (линейные операции) с помощью одной операции «слепого сдвига» LUT. Кроме того, за счет данной операции возможно реализовать и нелинейные операции, такие как например сравнение. Рассмотрим данную схему подробнее.

Особенностью данной схемы, является то, что она использует концентрированное распределение на торе $\mathbb{T}$. Основная идея заключена в том, что если распределение сосредоточено на небольшом интервале, тогда на торе возможно определить, как матожидание, так и дисперсию. Авторы утверждают, что распределение $\mathcal{X}$ на торе является концентрированным тогда и только тогда, когда его область входит в сферу, с радиусом $\frac{1}{4}$, вплоть до пренебрежимо малого, входящую в $\mathbb{T}$. В таком случае, дисперсия определяется, как $\mathrm{var}(\mathcal{X}) = \min_{\overline{x}\in\mathbb{T}} \sum \mathcal{X}|x - \overline{x}|^2$, а матожидание $\mathbb{E}(\mathcal{X})$ – как позиция $\overline{x} \in \mathbb{T}$, ко-

торая минимизирует выражение. Основываясь на этом, авторы утверждают, что матожидание $\mathbb{E}(\mathcal{X})$ – это вектор матожиданий каждого коэффициента, а $\mathrm{var}(\mathcal{X})$ обозначает максимум дисперсии каждого коэффициента. Эти матожидание и дисперсия над $\mathbb{T}$ подчиняются тем же правилам линейности, что и их классические эквиваленты над вещественными числами.

Для выполнения гомоморфной арифметики, авторы применяют свою версию обучения с ошибками: вместо OOK они используют обучение с ошибками в торе (OOT), при этом задействованы, как модифицированное тором OOK (OOKT), так и модификация тором внутреннего умножения GSW [172] (TGSW). В таком случае основные операции можно описать следующим образом:

- **Генерация секретного ключа**: секретный ключ s генерируется как случайная выборка из $\mathbb{T}^n$, где n – большое простое число;
- **Генерация публичного ключа**: задается элемент a, который получен из равномерного распределения по $\mathbb{T}^n$ и ошибка e – выбранная из распределения по $\mathbb{T}$, тогда открытый ключ вычисляется как $pk = (a, [a \cdot s + e]) = (p_0, p_1)$;
- **Шифрование**: задается элемент v, который получен из распределения по $\mathbb{T}$, из e берутся малые ошибки e_0 и e_1. Сообщение $\boldsymbol{\nu}$ шифруется, как $c = (v \cdot p_0 + e_0, \boldsymbol{\nu} + v \cdot p_1 + e_1) = (c_0, c_1)$;
- **Дешифрование**: чтобы извлечь сообщение из шифртекста, необходимо выполнить $\boldsymbol{\nu} = [c_1 - c_0 \cdot s]$.

Для реализации линейных операций используется TGSW с алгоритмом декомпозиции [177] $Dec_{H,\beta,\varepsilon}(b) \cdot A$, где A – первый множитель, b – второй множитель, $H \in M$ – гаджет представляющий собой блочно-диагональную матрицу, каждый блок столбцов которой содержит геометрически убывающую последовательность постоянных полиномов в $\mathbb{T} \subseteq \mathbb{T}N[X]$, принадлежащий модулю пространства OOK $\mathfrak{R} = \mathbb{Z}[X]/I$, где I – идеал $\mathbb{Z}[X]$, $\varepsilon = \mathbf{u} \cdot H - \mathbf{v}$, $\mathbf{u} \in \mathfrak{R}^{(k+1)\ell}$, где целое $k \geqslant 1$, ℓ – граница пространства, $\mathbf{v} \in \mathbb{T}_N[X]^{(k+1)}$, тогда:

$$\boxdot : TGSW \times \mathrm{OOT} \to \mathrm{OOT},$$
$$(A, b) \to A \boxdot b = Dec_{H,\beta,\varepsilon}(b) \cdot A.$$

Основой арифметики схемы TFHE являются операции слепого сдвига над LUT, для их реализации используется селекторный вентиль, или, как его называют авторы работы, вентиль CMux. Он имеет три входа и один выход. Два

входа предназначены для шифртекстов, ограниченных булевыми значениями, и один вход управления. Вход управления, как и входы шифртекстов, гомоморфно зашифрован. В зависимости от значения, подаваемого на вход управления, вентиль вращается в позицию с необходимой операцией. Операция получила название «слепой» по причине того, что как значения входных шифртекстов, так и операции, которые выполняет вентиль, конфиденциальны.

В следствие особенностей LUT, операции сдвига выполняются в TFHE с помощью булевых функций, таким образом, схема поддерживает только логические данные. Это обеспечивает высокую криптостойкость схемы [176] и скорость обработки данных. Кроме того LUT возможно использовать и для нелинейных операций [102], что позволяет расширять применение данной схемы.

1.4.2 Схема BFV использующая целочисленную арифметику

Схема BFV [75] поддерживает целочисленную арифметику с высокой точностью и большим количеством разрешенных операций умножения. Наравне со схемами CKKS и TFHE она является наиболее технологичной и популярной в научном сообществе. BFV, как и схема Gentry основана на OOK, поэтому пространство шифртекстов представляет из себя полиноминальное кольцо. Рассмотрим схему BFV подробнее. BFV описывается следующим набором основных параметров и обозначений:

- m – сообщение (до шифрования);
- n – степень модульного полинома;
- p – коэффициент модуля открытого текста;
- q – коэффициент модуля закрытого текста;
- Θ – набор параметров шифрования (n, p, q);
- Φ_n – циклотомический полином $(x^n + 1)$;
- $\mathcal{N}$ – гауссово распределение $\mathcal{N}(\mu,\sigma)$.

Тогда пространство открытых текстов представляет собой полиноминальное кольцо $R_p = \mathbb{Z}_p[x]/\Phi_n(x)$, а закрытых $R_q = \mathbb{Z}_q[x]/\Phi_n(x)$.

Кодирование в открытый текст выглядит следующим образом:

$$m_{end} = \text{encoding}(m) = c_{n-1}x^{n-1} + \ldots + c_1x^1 + c_0,$$

где c_i – коэффициенты полинома, $c_i \in \mathcal{N}/p$.

Шифрование производится за счет открытого ключа, который определяется как $k_p = \left(|-(ak_s + e)|_{\Phi_{n,q}}, a\right)$, где k_s – секретный ключ, a – многочлен n-й степени, коэффициенты которого выбираются случайным образом из множества $0, \ldots, q-1$, и e – многочлен n-й степени, коэффициенты которого выбираются случайным образом из дискретного и ограниченного гауссова распределения $\mathcal{N}(\mu = 0,\ \sigma = 3.2)$ над целыми числами. В таком случае шифрование представляется как

$$m_{enc} = E_\Theta(m_{end}, k_p) = \left(\left|k_p a + e_1 + \left\lfloor \frac{q}{p} \right\rfloor \cdot m_{end}\right|_{\Phi_{n,q}}, |k_p a + e_2|_{\Phi_{n,q}}\right).$$

Дешифрование же определяется следующей формулой

$$D_\Theta(m_{enc}, k_s) = m_{end} = \left|\left\lfloor \frac{p}{q} |m_{enc0} + m_{enc1} k_s|_{\Phi_{n,q}} \right\rceil\right|_p.$$

Теперь рассмотрим гомоморфные операции. В первую очередь рассмотрим сложение

$$m_{enc_1} + m_{enc_2} = \left(\left|m_{enc_{1_0}} + m_{enc_{2_0}}\right|_{\Phi_{n,q}}, \left|m_{enc_{1_1}} + m_{enc_{2_1}}\right|_{\Phi_{n,q}}\right).$$

Гомоморфное умножение описывается следующим образом

$$m_{enc_1} *\ m_{enc_2} = \begin{pmatrix} \frac{p}{q} \left|m_{enc_{1_0}} m_{enc_{2_0}}\right|_{\Phi_{n,q}}, \\ \frac{p}{q} \left|m_{enc_{1_0}} m_{enc_{2_1}} + m_{enc_{1_1}} m_{enc_{2_0}}\right|_{\Phi_{n,q}}, \\ \frac{p}{q} \left|m_{enc_{1_1}} m_{enc_{2_1}}\right|_{\Phi_{n,q}}. \end{pmatrix}.$$

Таким образом выполняются операции над зашифрованными данными в схеме BFV. Для повышения быстродействия используются такие операции как релинеаризация [134] и бутстраппинг [83]. С помощью гомоморфного сложения и умножения возможно реализовать и другие операции: вычитание, возведение в степень и т.п. В случае более нетривиальных операций, таких как операции матричной алгебры, возникают определенные сложности. Схема BFV имеет достаточно высокую криптостойкость [159].

1.4.3 Схема CKKS построенная на рациональной арифметике с фиксированной запятой

Следующим шагом в истории ПГШ стала схема CKKS (первоначально HEaaN) [13], которая позволяет обрабатывать рациональные числа [13]. CKKS – это схема гомоморфного шифрования, предназначенная для эффективного выполнения приближенных арифметических операций над зашифрованными данными. Она идеально подходит для вычислений с вещественными или комплексными числами над полем $C^{N/2}$. Пространства открытых текстов и шифртекстов совпадают

$$Z_Q[X]/(X^N + 1), \tag{1.3}$$

где N – обычно степень двойки. Пакетное кодирование $C^{\frac{N}{2}} \leftrightarrow Z_Q[X]/(X^N + 1)$ отображает массив комплексных чисел в многочлен со свойством: $\text{decode}(\text{encode}(m_1) \otimes \text{encode}(m_2)) \approx m_1 \odot m_2$, где $\otimes$ – покомпонентное умножение, а $\odot$ – негациклическая свертка.

Схема CKKS работает по стандарту [97], который содержит рекомендуемые параметры для 128-битных ключей ПГШ троичной формы $s \in -1{,}0{,}1)^N$. Шифрование в CKKS осуществляется путем вычисления полиномов Лагранжа в поле комплексных чисел.

Схема использует приближенную арифметику для построения шифртекстов. Рассмотрим заданную арифметику. В начале фиксируем основание $p > 0$ и модуль q_0, причем $q = p \cdot q_0$ при $0 < l \leqslant L$. Целое число p будет использоваться в качестве основы для масштабирования в приблизительных расчетах. В качестве параметра безопасности λ такое, что главный модуль $M = M(\lambda, q_L)$ для полиномиального кольца. При границах $0 < l \leqslant L$ уровня шифртекста l определяется вектор в $\mathcal{R}_{q_l}^k$ для фиксированного целого числа k.

Рассмотрим основные операции схемы CKKS:

Генерация ключей: процесс шифрования начинается с генерации открытого ключа pk и закрытого ключа sk. Закрытый ключ используется для дешифровки данных, а открытый – для их шифрования.

Шифрование: чтобы зашифровать вектор открытого текста x, выполняются следующие действия:

Кодирование: вектор открытого текста x кодируется в полином открытого текста $m(x)$, который является полиномиальным представлением сообщения;

Дополнение: вектор $m(x)$ дополнен нулями, длина вектора равна заданной степени двойки N;

Гомоморфное шифрование: полином $m(x)$ шифруется с помощью pk для получения полинома $c(x)$ шифртекста, при этом контролируется уровень шума шифртекста – количество специально вносимых ошибок e, удовлетворяющее неравенству $|e|_{\infty}^{can} \leqslant e_{Max}$, где $|e|_{\infty}^{can}$ – каноничная норма вложения и e_{Max} – заданная константа, для выражения $\langle c, sk\rangle = m + e_{Max} \,(\mathrm{mod} q_L)$.

Дешифрование: для дешифровки полинома $c(x)$ шифртекста выполняются следующие действия:

Гомоморфное дешифрование: полином $c(x)$ дешифруется с помощью секретного ключа для получения полинома $m(x) \leftarrow \langle c, sk\rangle \,(\mathrm{mod} q_l)$ в пространстве открытых текстов;

Декодирование: для получения исходного текстового вектора x текстовый полином $m(x)$ снова преобразуется из полинома в полином сообщений.

CKKS поддерживает несколько приближенных арифметических операций над зашифрованными данными, включая сложение и умножение. Гомоморфное сложение и умножение можно выполнять в пространстве шифртекстов без необходимости их дешифровки.

Гомоморфное сложение: при получении двух шифртекстов $c_1(x)$ и $c_2(x)$, представляющих зашифрованные значения $m_1(x)$ и $m_2(x)$ соответственно, выполняется гомоморфное сложение путем сложения соответствующих коэффициентов по модулю: $c(x) = c_1(x) + c_2(x)$, при этом ошибки e_1 и e_2 также суммируются.

Гомоморфное умножение: при наличии двух шифртекстов $c_1(x)$ и $c_2(x)$, представляющих зашифрованные значения $m_1(x)$ и $m_2(x)$ соответственно, гомоморфное умножение выполняется путем преобразования зашифрованных полиномиальных текстов для последующего покомпонентного умножения по модулю исходного модульного текста и обратного преобразования: $c\,(x) = c_1\,(x) \times c_2(x)$, для умножения выделяются собственные границы ошибок $e_{mult} \in \mathcal{R}$ с $|e_{mult}|_{\infty}^{can} \leqslant e_{mult_{Max}}$.

Как сложение, так и умножение приводят к увеличению ошибки аппроксимации e, схема CKKS позволяет дешифровать данные, если ошибка находится

в определенных пределах. При использовании схемы CKKS важно контролировать рост ошибки, который зависит от количества операций и их порядка. Учитывая особенности арифметики, умножение вносит большую погрешность. Различные программные реализации схемы CKKS предлагают разные способы контроля уровня ошибки. Для ускорения арифметики CKKS используется система остаточных классов, что так же открывает новые возможности для модификации данной схемы ПГШ [70].

1.5 Заключение по первой главе

В первой главе представлен анализ проблемы обеспечения конфиденциальности искусственного интеллекта в публичных распределенных системах. Рассмотрены приложения искусственного интеллекта в распределенных системах. Установлено, что методы искусственного интеллекта эффективно работают в связке с ОВ и ТВ, но ОВ и ТВ имеют проблемы безопасности в публичных сетях. Анализ проблем безопасности и путей их решения показал, что ПГШ позволяет обрабатывать данные в зашифрованном виде с помощью операций гомоморфного сложения и умножения.

НС, как метод искусственного интеллекта, в настоящий момент активно развиваются научным сообществом и представляют большой интерес. Установлено, что НС возможно реализовать с помощью гомоморфных операций. Таким образом, для дальнейших исследований было принято решение рассматривать НССК на основе ПГШ. Аналитический обзор ПГШ показал, что наибольший интерес представляют схемы TFHE, BFV и CKKS. CKKS поддерживает гомоморфную арифметику над вещественными числами, BFV является достаточно точной целочисленной схемой, а TFHE является наиболее безопасной и эффективной схемой, в которой LUT позволяет реализовать в том числе и нелинейные операции посредством логических сдвигов.

Проанализированные работы, в которых рассмотрено применение ПГШ для реализации НССК, содержат требующие дополнительного исследования и обсуждения результаты: с одной стороны, НССК позволяют обрабатывать данные с высоким уровнем конфиденциальности, с другой стороны, такие НССК требуют большое количество вычислительных ресурсов. Учитывая все

вышесказанное, следующий этап исследований посвящен разработке методов и алгоритмов повышения эффективности обработки данных в НССК с ПГШ.

Глава 2. Разработка методов и алгоритмов нейронных сетей сохраняющих конфиденциальность

2.1 Библиотеки полностью гомоморфного шифрования

ПГШ активно развивается, для проведения практических исследований несколькими группами были разработаны библиотеки ПГШ [3]. Библиотеки можно разделить по языкам разработки, основные и наиболее популярные библиотеки реализованы на языке C++, некоторые из них были портированы на Python [99; 100]. Имеются так же реализации на Cuda [58] и GoLang [113]. Подробно остановимся на библиотеках, написанных на C++.

Одной из первых библиотек является библиотека HElib [92], разработанная IBM. Изначально библиотека поддерживала схему BGV, которая была разработана в IBM, позднее так же была интегрирована библиотека CKKS. CKKS имеет оригинальную библиотеку, которая носит название HEaaN [94] (такое же как и оригинальное название самой схемы CKKS). Разработчики TFHE представили свою библиотеку [176]. OpenFHE [133] развилась из PALISADE и стала крупной библиотекой по работе с ПГШ. Главным ее преимуществом является более дружественное к пользователю применение ПГШ, почти все параметры и операции проводятся и вычисляются самой библиотекой, от пользователя требуется только определение базовых параметров и соблюдение правил и требований по работе с ПГШ. Рассматриваемые библиотеки являются open-source проектами и зачастую представляют либо какую-то уникальную технологию (как TFHE), либо конкретную схему (как HEaaN), исключением является OpenFHE. Крупные компании, такие как IBM, часто интегрируют другие библиотеки и создают свой API, как, например, GoogleFHE [89], который включает TFHE и OpenFHE, либо создают обширную библиотеку, интегрируя целый набор библиотек, а также технологий по работе с ПГШ, как поступили Microsoft, создав библиотеку SEAL [121].

SEAL требует более тонкой настройки параметров ПГШ и последовательного выполнения гомоморфных операций с контролем накопления ошибки и размера ключей. Но несмотря на сложности в работе, SEAL предоставляет функционал для точных и быстрых ПГШ вычислений, благодаря чему SEAL

одной из первых была портирована на Python. Реализация SEAL на Python была ожидаемым шагом: учитывая то, как работает ПГШ, его применение для НС было вопросом времени. Python обладает большим количеством библиотек для работы с НС. Одной из таких библиотек является PyTorch [51], помимо инструментария для работы с НС, она также поддерживает тензорные вычисления. На основе PyTorch и SEAL была разработана библиотека TenSEAL [174], которая совмещает векторную обработку шифртекстов SEAL и тензорные вычисления PyTorch. Данная библиотека дает возможность пользователю достаточно просто разрабатывать СНССК (Сверточные НССК). Совмещение тензоров и гомоморфного шифрования дает достаточно хорошие результаты с точки зрения скорости обработки данных, кроме того, авторы TenSEAL произвели модификацию гомоморфных операций, что позволило проводить их еще эффективнее. Авторами была опубликована работа, показывающая возможности библиотеки. Таким образом, для исследований СНССК в рамках данной работы была выбрана библиотека TenSEAL.

Для работы с TFHE, помимо одноименной библиотеки на C++, существует библиотека Concrete [56]. Данная библиотека представляет собой инструментарий для работы с TFHE на Python, так же имеет реализацию на языке Rust. Однако, для данного исследования более интересен расширенный вариант библиотеки Concrete, а именно Concrete-ML [184]. Concrete-ML предоставляет инструментарий для работы с методами МО, в том числе с НС и СНС, на базе TFHE. Библиотека содержит набор уже разработанных и оптимизированных для работы с TFHE моделей искусственного интеллекта, а так же инструменты для разработки пользовательских моделей, что важно при исследовании СНССК. В набор инструментов входят функции для работы с операциями над LUT, квантизацией, а так же шифрованием и дешифрованием моделей на базе TFHE.

Таким образом, для исследования схем CKKS и BFV будет использоваться библиотека TenSEAL, а для исследования схемы TFHE библиотека Concrete-ML.

Определившись с основным математическим аппаратом, перейдем к разработке модели СНС, сохраняющей конфиденциальность. Как уже отмечалось выше, выбранные ПГШ схемы сохраняют результат сложения и умножения над шифртекстами, что обеспечивает корректное выполнение свертки. Одна-

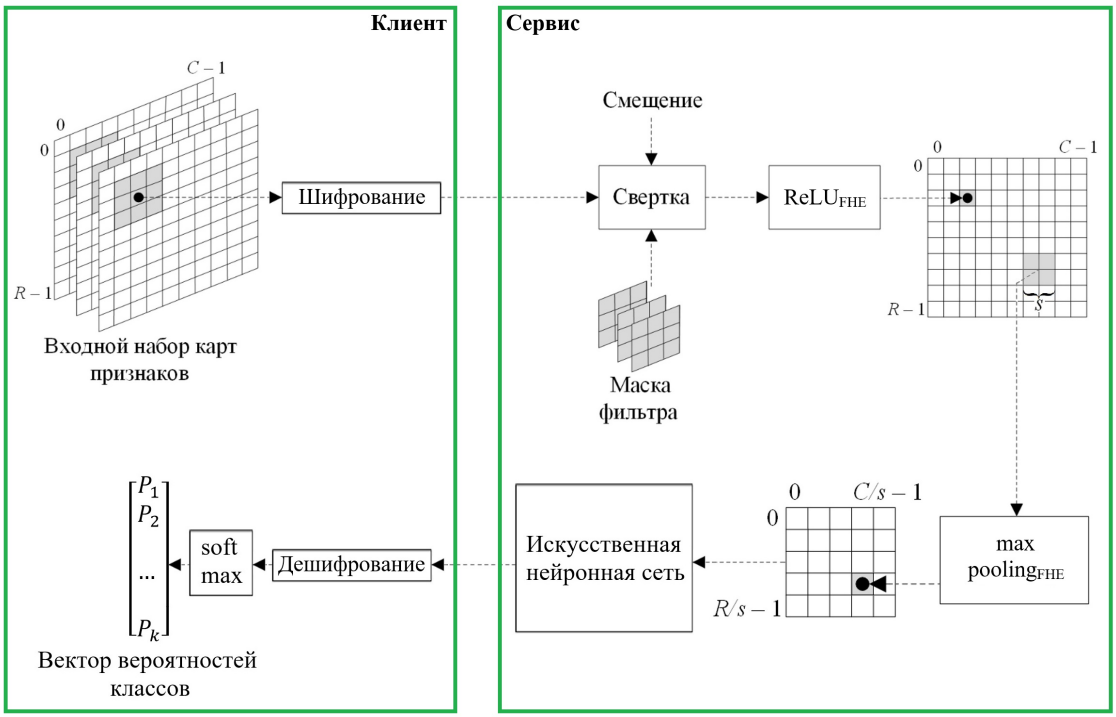


Рисунок 2.1 — Архитектура CNN для удаленной обработки конфиденциальных данных

ко, остается немало операций СНС, «не вписывающихся» в данную концепцию: определение знака (ReLu), возведение экспоненты в степень и деление (softmax).

Начнем с конца. Проблема с реализацией функции softmax решается предварительным дешифрованием с последующим выполнением функции softmax уже на стороне клиента (рис. 2.1).

Таким образом, обходится проблема гомоморфного возведения экспоненты в степень и гомоморфного деления, но остаются проблемы с определением знака зашифрованного числа для реализации функции ReLU над шифртекстами (на рис. 2.1 ReLU_{FHE}). В данном случае, для того, чтобы избавиться от операции определения знака числа, возможно применение численных или приближенных методов для получения полиномиального приближения ReLU, либо применение функций активации не требующих операции определения знака числа, что будет рассмотрено позднее.

2.2 Повышение эффективности умножения на зашифрованную матрицу

Умножение матриц – базовая операция для многих систем, в том числе и для НС. Рассмотрим ее алгоритм, основанный на умножении квадратных матриц a и b размера $n \times n$:

$$c_{i,j} = \sum_{k=1}^{n} a_{i,k} \cdot b_{k,j},$$

где $i, j, k \in \overline{1, n}$, c – результат умножения. В открытом виде этот алгоритм довольно прост. Однако в ПГШ его выполнение невозможно, так как мы не можем отдельно обратиться к элементу каждого внутреннего вектора. Рассмотрим подробно метод Halevi и Shoup, а так же алгоритм Jiang и др. [155]. Алгоритм Halevi и Shoup [93] основан на последовательности умножений матрицы на вектор. Он кодирует каждый вектор матрицы как открытый текст, т.е. создается открытый текстовый вектор. Затем закодированная матрица шифруется как вектор шифртекстов. Наконец, вектор шифртекстов шифруется в единый шифртекст. Операции выполняются с этим шифртекстом.

При умножении матрицы на вектор входные матрицы кодируются в их диагональном представлении, т.е. каждая диагональ кодируется в зашифрованный текст.

Пусть матрица A размера $n \times n$ задана как $a_0, \ldots, a_{n-1}$, где $a_i = (A_{0,i}, A_{1,i+1}, \ldots, A_{n-1,n+i-1})$. Следовательно, $a_i[j] = A_{j,j+i}$. Произведение $w = vA$, где v входной вектор, можно вычислить по формуле $w \leftarrow \sum_{i=0}^{n-1} a_i \times (v \lll i)$.

Этот метод требует n вращений, умножений и сложений. Мультипликативная глубина равна 1. Алгоритм Jiang и др. [155] основан на линейном преобразовании квадратных матриц.

Для матрицы $U \in \mathcal{R}^{n \times n}$, линейное преобразование $L : \mathcal{R}^n \to \mathcal{R}^n$ может быть представлено как $L : m_e \longmapsto U \cdot m_e$. Таким образом, умножение матрицы на вектор можно выполнить, объединив операции поворота и умножения на константу.

Для $0 \leqslant \ell < n$, ℓ-й диагональный вектор U можно определить как

$$u_\ell = (U_{0,\ell,U1,\ell+1},...,U_{n-\ell-1,n\breve{}1},U_{n-\ell,0,...,Un\breve{}1,\ell-1}) \in \mathcal{R}^n,$$
$$U \cdot m_e = \sum_{0 \leqslant \ell < n} (u_\ell \odot \rho(m_e; \ell)),$$

где $\odot$ это покомпонентное умножение векторов.

$A = (A_{i,j})_{0 \leqslant i,j < n}$ – матрица размера $n \times n$. Перестановки σ, τ, φ и ψ на множестве $\mathcal{R}^{n \times n}$ определяются следующим образом

$$\sigma(A)_{i,j} = A_{i,i+j};$$
$$\tau(A)_{i,j} = A_{i+j,j};$$
$$\varphi(A)_{i,j} = A_{i,j+1};$$
$$\psi(A)_{i,j} = A_{i+1,j}.$$

Умножение матриц можно задать следующей формулой

$$A \cdot B = \sum_{k=0}^{n-1} (\varphi^k \circ \sigma(A)) \odot (\psi^k \circ \tau(B)), \tag{2.1}$$

где $\circ$ обозначает композицию функций. Алгоритм умножения требует определения матричных представлений, соответствующих перестановкам U^σ, U^τ, V^k и W^k. Для $0 \leqslant i, j < n$, $1 \leqslant k < n$ и $0 \leqslant \ell < n^2$:

$$U^\sigma_{n \cdot i+j,\ell} = \begin{cases} 1, & \text{если } \ell = n \cdot i + |i+j|_n \\ 0, & \text{в противном случае} \end{cases}; \tag{2.2}$$

$$U^\tau_{n \cdot i+j,\ell} = \begin{cases} 1, & \text{если } \ell = n \cdot |i+j|_n + j \\ 0, & \text{в противном случае} \end{cases}; \tag{2.3}$$

$$V^k_{n \cdot i+j,\ell} = \begin{cases} 1, & \text{если } \ell = n \cdot i + |j+k|_n \\ 0, & \text{в противном случае} \end{cases}; \tag{2.4}$$

$$W^k_{n \cdot i+j,\ell} = \begin{cases} 1, & \text{если } \ell = n \cdot |i+k|_n + j \\ 0, & \text{в противном случае} \end{cases}. \tag{2.5}$$

Умножение двух зашифрованных матриц $ct.A$ (шифртекст A) и $ct.B$ (шифртекст B) выполняется следующим образом.

Шаг 1.1. Линейное преобразование U^σ с $ct.A$ на входе:

$$U^\sigma \cdot a = \sum_{-n < k < n} \left(u^\sigma_k \odot \rho(a; k)\right), \tag{2.6}$$

где $a = i^{-1}(A) \in \mathcal{R}^n$ является векторным представлением A. Это выражение можно вычислить в схеме ПГШ как

$$\sum_{-n<k<n} \mathrm{CMult}(\mathrm{Rot}(ct.A; k); u_k^{\sigma}). \tag{2.7}$$

Шаг 1.2. Линейное преобразование U^{τ} с $ct.B$ на входе:

$$U^{\tau} \cdot b = \sum_{0 \leqslant k<n} \left(u_{n \cdot k}^{\tau} \odot \rho\left(b; n \cdot k\right)\right),$$

где $b = i^{-1}(B) \in \mathcal{R}^n$ и $u_{n \cdot k}^{\tau}$ является диагональным вектором U^{τ}.

В ПГШ это представлено следующим образом

$$\sum_{0 \leqslant k<n} \mathrm{CMult}(\mathrm{Rot}(ct.B; n \cdot k); u_{n \cdot k}^{\tau}).$$

Шаг 2. Гомоморфное вычисление операций $\sigma(A)$ и $\tau(B)$. Для $1 \leqslant n < d$ матрица сдвига столбцов V^k содержит два ненулевых диагональных вектора v_k и v_{k-n}:

$$\begin{aligned} \mathbf{v}_k[\ell] &= \begin{cases} 1, & \text{если } 0 \leqslant |\ell|_n < (n-k) \\ 0, & \text{в противном случае} \end{cases}; \\ \mathbf{v}_{k-n}[\ell] &= \begin{cases} 1, & \text{если } (n-k) \leqslant |\ell|_n < n \\ 0, & \text{в противном случае} \end{cases}. \end{aligned} \tag{2.8}$$

Сложив два шифртекста $\mathrm{CMult}(\mathrm{Rot}(ct.A^{(0)}; k); v_k)$ и $\mathrm{CMult}(\mathrm{Rot}(ct.A^{(0)}; k-n); v_{k-n})$ получаем $ct.A^{(k)}$ и $ct.B^k \leftarrow \mathrm{Rot}(ct.B^{(0)}; n \cdot k)$. Этот шаг требует n сложений, $2n$ умножений на константу и $3n$ вращений.

Шаг 3. На этом шаге выполняется умножение Hadamard для шифртекстов $ct.A^{(k)}$ и $ct.B^{(k)}$ для $0 \leqslant k < n$, и получается результирующий шифртекст. Этот шаг требует n гомоморфных сложений и умножений.

Хотя метод, предложенный Jiang и др., требует большей глубины по сравнению с методом Halevi и Shoup, алгоритм имеет меньшую временную и пространственную сложность (количество шифртекстов). Эти факторы имеют решающее значение для эффективной реализации нейронных сетей сохраняющих конфиденциальность, особенно при обработке больших объемов данных.

Если рассматривать пространственную сложность, то для данного алгоритма она составляет $O(n^4)$.

Однако есть и другой способ, который подходит для СНССК. Возникает вопрос, настолько ли необходим подобный уровень конфиденциальности. Учитывая тот факт, что в настоящее время невозможно обучить СНС в ПГШ за приемлемое время, СНС обучается в открытом виде. Справедливо заметить, что значения весов являются открытыми, могут быть общедоступными и существует возможность их компрометации. В таком случае необходимость их шифрования минимальна, и возможно их применение в открытом виде. Тогда можно использовать модифицированный алгоритм, основанный на предыдущем, где входная матрица кодируется в диагональном представлении, а веса представляются в виде вектора. В таком случае формула (2.1) изменит свой вид. Рассмотрим теорему.

Теорема 1. *Если матрица A зашифрована, криптографической полностью гомоморфной схемой, а матрица B представлена в открытом виде, тогда*

$$A \cdot B = \sum_{k=0}^{n-1} (\varphi^k \circ \sigma(A)) \odot (\psi^k \circ \tau(B)),$$

принимает вид умножения матрицы на скаляр

$$A \cdot B = \sum_{k=0}^{n-1} (\varphi^k \circ \sigma(A)) \odot \left(B^{k^T}\right), \tag{2.9}$$

где B^{k^T} компонентный вектор транспонированной матрицы B. В таком случае количество операций сложения составляет n^2, количество операций умножения n^2, а количество операций вращения n, а сложность уменьшается с $O(n^4)$ до $O(n^2)$.

Доказательство. Рассмотрим алгоритм, описанного выше, но с поправкой на (2.9). На подготовительном этапе вычисляются вспомогательные матрицы по формулам (2.2), (2.3).

Шаг 1.1. Линейное преобразование U^σ с $ct.A$ на входе по формулам (2.6), (2.7).

Шаг 1.2. Получение транспонированной матрицы B, т.е. вычисление B^T.

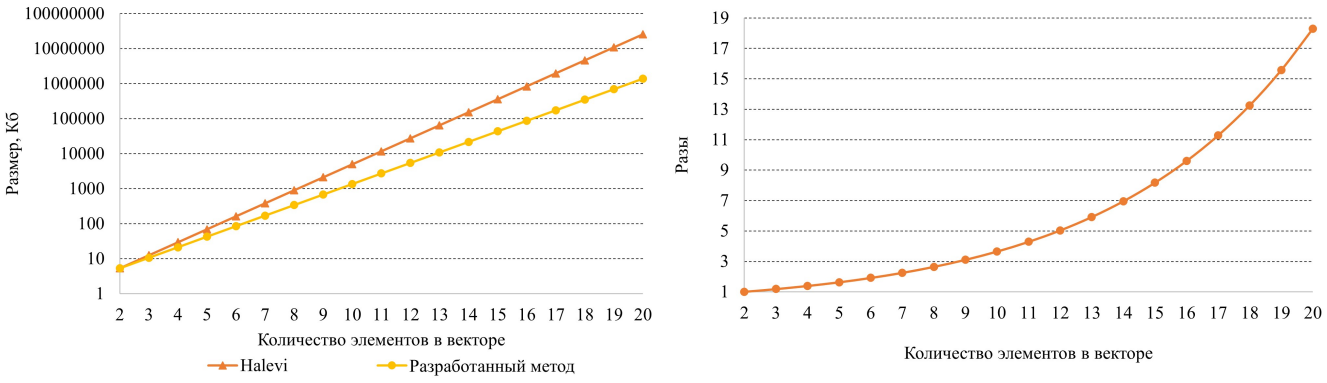
Шаг 2. Гомоморфное выполнение операций $\sigma(A)$. Для $1 \leqslant k < n$ матрица сдвига столбцов V^k содержит два ненулевых диагональных вектора v_k и v_{k-n}, вычисляемых по формуле (2.8).

Шаг 3. Вычисляется скалярное умножение шифртекстов $ct.A^{(k)}$ на открытые тексты ${B^{(k)}}^T$ для $0 \leqslant k < n$, и получается результирующий шифртекст. Скалярное умножение шифртекстов требует n гомоморфных сложений и умножений.

Таким образом, за счет представления матрицы B в открытом виде операция умножения $ct.A^{(k)}$ на ${B^{(k)}}^T$ не требует вычисления вспомогательных матриц для матрицы B, что позволяет сократить количество умножений и вращений, однако, необходима одна дополнительная операция вращения, для проведения операции транспонирования.

Теорема доказана. □

Этот метод требует n операций вращения, и по n^2 операций умножения и сложения. Кроме того, вспомогательные матрицы (2.2)-(2.5) в зашифрованном виде занимают много памяти, как и промежуточные зашифрованные матрицы, что было показано в работе [17]. Из формул видно, что, используя открытые значения, можно сократить не только количество операций, но и расход памяти. Для подтверждения выводов было проведено экспериментальное исследование (рис. 2.2).



а) б)

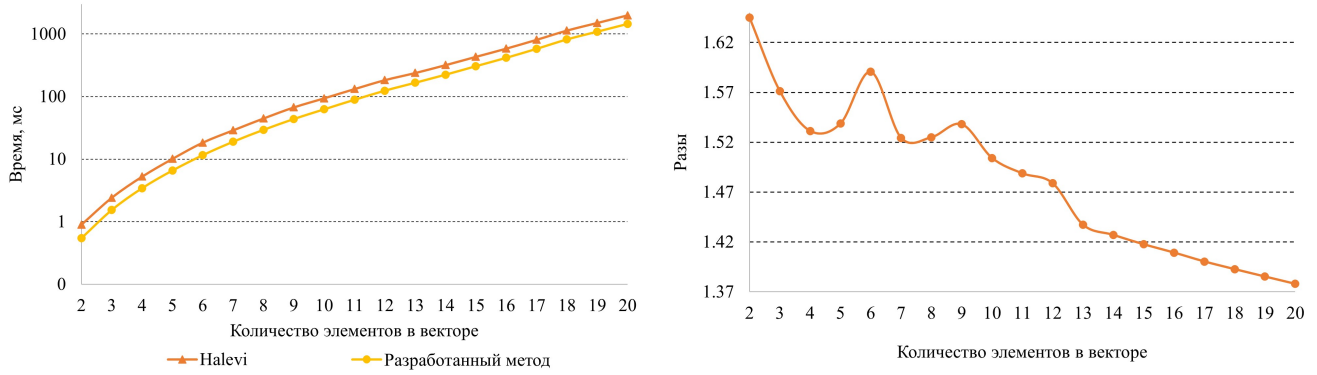
Рисунок 2.2 — Экспериментальное сравнение метода Halevi с предлагаемым с точки зрения потребляемой памяти: а) потребляемая память; б) отношение потребляемой памяти

Из данных, представленных на Рисунке 2.2а), можно сделать вывод, что предложенный алгоритм позволяет сократить объем потребляемой памяти в среднем в 7.89 раза. Пространственная сложность при этом уменьшилась с $O(n^4)$ до $O(n^2)$ для произведения матриц размера $n \times n$. Учитывая, что для произведения квадратных матриц размера $n \times n$ необходимо выполнить n^2

скалярных умножений и $2n^2$ циклических сдвигов векторов, пространственная сложность алгоритма скалярного умножения на зашифрованные данные снижается с $O(n^2)$ до $O(n)$. Как видно из Рисунка 2.2б), потребление памяти сократилось с квадратичного роста до линейного. Это дает преимущество в эффективности при работе с CHCCK. Однако, учитывая специфику схемы CKKS, необходимо проверить, не повлияли ли внесенные изменения на точность результата. Для этого необходимо построить нейронную сеть и провести исследование.

Далее оценим скорость, с которой выполняются вычисления. Стоит отметить, что на Рисунке 2.3 а) показана общая скорость выполнения операций, включая шифрование и дешифрование.

Анализируя Рисунок 2.3а), можно сказать, что предложенный метод выполняет умножение быстрее. Этот эффект достигается как за счет того, что шифрование и дешифрование упрощаются, поскольку умножение выполняется на открытый вектор весов CHCCK. На Рисунке 2.3 б) представлено соотношение скоростей сравниваемых методов.



а) б)

Рисунок 2.3 — Экспериментальное сравнение метода Halevi с предлагаемым с точки зрения производительности: а) время выполнения; б) отношение времени выполнения

Линии тренда зависимости времени от n для алгоритма Halevi $0.0634n^4 - 1.8561n^3 + 20.897n^2 - 88.794n + 118.56$, для предложенного алгоритма $0.0461n^4 - 1.3405n^3 + 14.955n^2 - 63.491n + 84.81$ с коэффициентом детерминации для обеих линий равным $R^2 = 0.9995$ (рис. 2.3а)). Асимптотически выигрыш во времени при увеличении n равен:

$$\lim_{n\to\infty} \frac{0.0634n^4 - 1.8561n^3 + 20.897n^2 - 88.794n + 118.56}{0.0461n^4 - 1.3405n^3 + 14.955n^2 - 63.491n + 84.81} \approx 1.38. \qquad (2.10)$$

Время работы алгоритма произведения квадратных матриц $n \times n$ уменьшилось в среднем в 1.49 раза (рис. 2.3б)). С увеличением размера матриц график становится более линейным, что можно объяснить увеличением избыточности, которая зависит от длины вектора, в то время как при малых размерах она зависит как от конструкции вектора, так и от вспомогательных матриц, необходимых для вращения.

В целом можно сказать, что предложенный метод эффективен как с точки зрения потребления памяти, так и скорости вычислений. Стоит отметить, что этот результат достигается за счет снижения конфиденциальности, а именно конфиденциальности весов СНССК, при условии, что веса являются общеизвестными по причине обучения СНССК в открытом виде.

В рамках исследования были проведены эксперименты по обучению и тестированию нейронной сети, а также ее зашифрованной версии на наборе данных MNIST. Целью эксперимента было оценить производительность модели в обычном и зашифрованном режимах, а также изучить влияние гомоморфного шифрования на производительность и точность модели. Рассмотрим СНС со следующими слоями и параметрами:

- Входное изображение: I, одноканальное изображение.
- Первый сверточный слой (C_1): применяет 4 фильтра размером 7×7 с шагом 3 (как по ширине, так и по высоте изображения) и размером рамки 0.
- Первый полносвязный слой (F_1): преобразует упрощенные карты признаков с использованием N нейронов.
- Второй полносвязный слой (F_2): сопоставляет скрытый слой с выходным слоем с использованием O нейронов.

Математические операции, выполняемые СНС, выглядят следующим образом:

- Работа первого сверточного слоя может быть определена как:
  - $C_1(I) = Conv2d(I, K_i, S_1, P_1)$, где K_i – фильтры с размером ядра 7×7, $i = 1..4$, с шагом $S_1 = 3$ и размером рамки $P_1 = 0$.
  - Выходной сигнал C_1 проходит через функцию активации и, возможно, другие операции, такие как объединение или нормализация, после чего сглаживается и поступает на первый полносвязный слой.
- Работа первого полносвязного слоя может быть определена как:

- $F_1(X) = XW_{F_1} + b_{F_1}$, где X – входной вектор для F_1, W_{F_1} и b_{F_1} представляют собой веса и смещения F_1, соответственно.

– Работа второго полносвязного слоя аналогично определяется как:

- $F_2(Y) = YW_{F_2} + b_{F_2}$, где Y – входной вектор F_2, полученный из выхода F_1, W_{F_2} и b_{F_2} веса и смещения F_2, соответственно.

Эта модель описывает структуру СНС, подчеркивая последовательность от обработки на сверточном слое до генерации конечного вывода через полносвязные слои. СНС была выбрана в качестве модели потому, что ПГШ обладает свойствами как гомоморфизма, так и автоморфизма, благодаря которым вращение зашифрованных матриц для реализации матричного умножения реализовать достаточного просто. Кроме того, эти свойства позволяют достаточно эффективно выполнять операцию свертки. Далее рассмотрим результаты работы полученной модели.

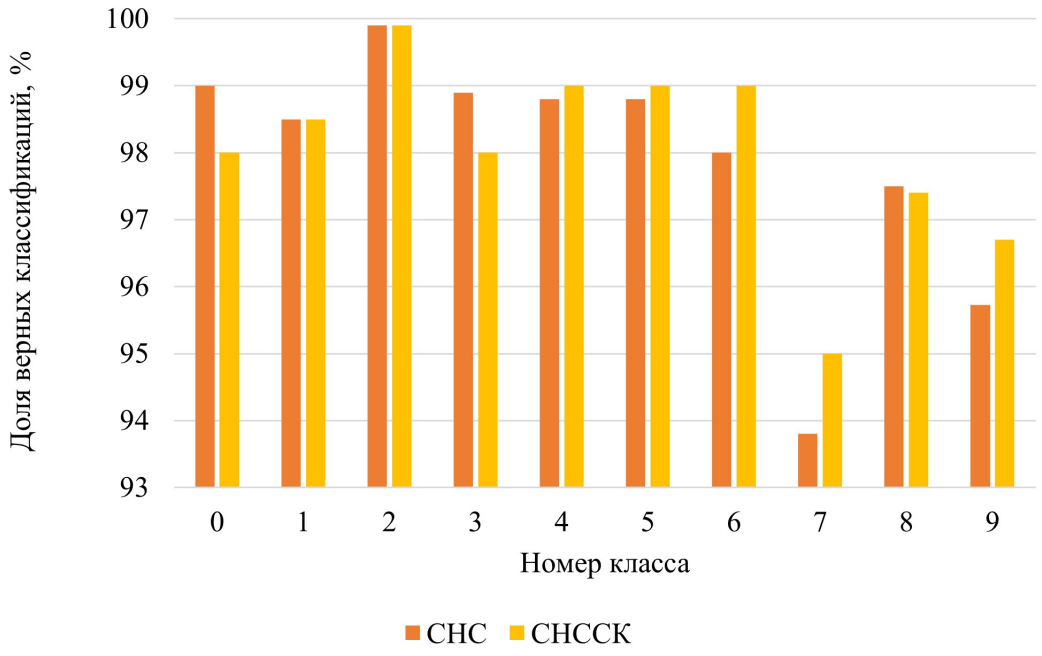


Рисунок 2.4 — Экспериментальное сравнение точности СНС и СНССК на наборе MNIST

На Рисунке 2.4 показана точность классификации на тестовых данных для каждого из 10 классов. По оси абсцисс представлены классы (от 0 до 9), а по оси ординат – доля верных классификаций для каждого из классов. График позволяет понять, как модель справляется с классификацией различных категорий, и выявить классы, в которых модель работает лучше или хуже. Результаты экспериментов показывают, что нейронная сеть демонстрирует высокую точность как в обычном, так и в зашифрованном режимах, причем в

зашифрованном режиме общая точность несколько повышается. Это говорит о том, что применение гомоморфного шифрования не оказывает существенного негативного влияния на способность модели к классификации.

Несмотря на то, что модификация метода умножения позволила повысить эффектность СНССК за счет снижения потребления памяти и повышения скорости умножения матриц, этого недостаточно. Рассматриваемая СНС имела небольшой размер, именно поэтому стала возможной реализация ее зашифрованной версии. Для повышения эффективности требуется повысить точность СНССК и сократить потребление вычислительных ресурсов, для этого в исследовании рассматриваются такие методы, как дистилляция и квантизация СНС.

2.3 Исследование методов дистилляции сверточных нейронных сетей

2.3.1 О методах дистилляции

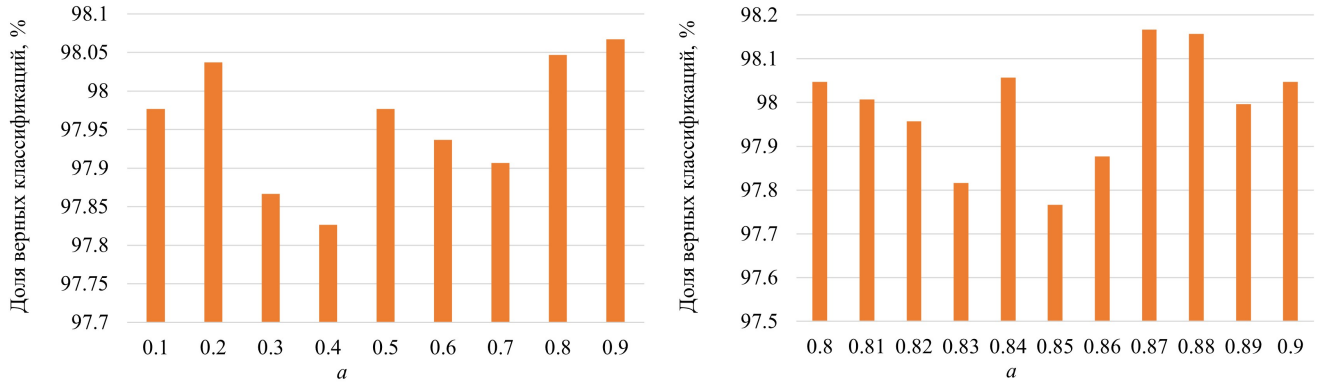
Решающим фактором, который влияет на точность любой СНС, является ее размер. Чем сложнее задача, тем большее количество нейронов необходимо для получения наиболее точного ответа. Конечно, свою роль играют и параметры сети, метод обучения и данные, на которых сеть обучается, но строго говоря, например, языковые модели требуют применения СНС достаточно больших размеров для эффективной работы. Кроме того, стоит учитывать и условия применения уже обученной сети, если будут использоваться маломощные устройства, то большая нейронная сеть не сможет эффективно работать. В таких случаях используется дистилляция нейронных сетей. Ее особенностью является то, что для малой СНС в качестве учителя выступает большая СНС, обладающая высокой точностью. Такое обучение позволяется получить точность малой СНС достаточно близкую к точности большой СНС, либо, в некоторых случаях, даже такую же. Рассмотрим данный процесс подробнее.

Дистилляция, как процесс обучения нейронной сети с учителем, впервые была представлена Tang и др. в работе [64]. Ключевая идея заключается в том, что при расчете квадратов расстояний до целевых значений из формулы (1.2),

так же учитываются и расстояния до решений «учителя», в качестве которого выступает обученная более мощная СНС. Тогда (1.2) преобразуется в следующее выражение

$$E(w_{i,j}) = a \cdot E_{CE} + (1 - a) \cdot E_{distill}, \tag{2.11}$$

где E_{CE} – расстояние от ответа СНС ученика до целевого значения, $E_{distill}$ – расстояние от ответа СНС ученика до ответа СНС учителя, $a \in [0,1]$ – константа. С приближением параметра a к 1, процесс обучения СНС ученика будет в большей степени ориентирован на целевые значения обучающей выборки, а значит будет нацелен на повторение процесса обучения СНС учителя, но в условиях ограниченности размеров сети. Если СНС ученика уступает размерам СНС учителя существенно, такой выбор параметра a может привести к резкому снижению точности. С приближением же параметра a к 0, процесс обучения СНС ученика будет в большей степени ориентирован на результаты СНС учителя, которые уже содержат ошибку. Таким образом, СНС ученика будет решать по-сути не исходную задачу, а какую-то новую, похожую на исходную, что с точки зрения логики должно отрицательно сказаться на точности, но при этом дает возможность (но не гарантию) обучить СНС ученика меньшего размера без резкой потери точности. Оптимальное значение параметра a определяется экспериментально, так же как, например, коэффициент регуляризации во многих алгоритмах МО (рис. 2.5).



а) б)

Рисунок 2.5 — Зависимость качества обучения СНС с дистилляцией от величины константы a: а) От 0.1 до 0.9; б) от 0.8 до 0.9.

Кроме того, для повышения качества дистилляции можно использовать коалицию СНС учителей. Оптимальный размер коалиции так же был определен в ходе эксперимента (рис. 2.6).

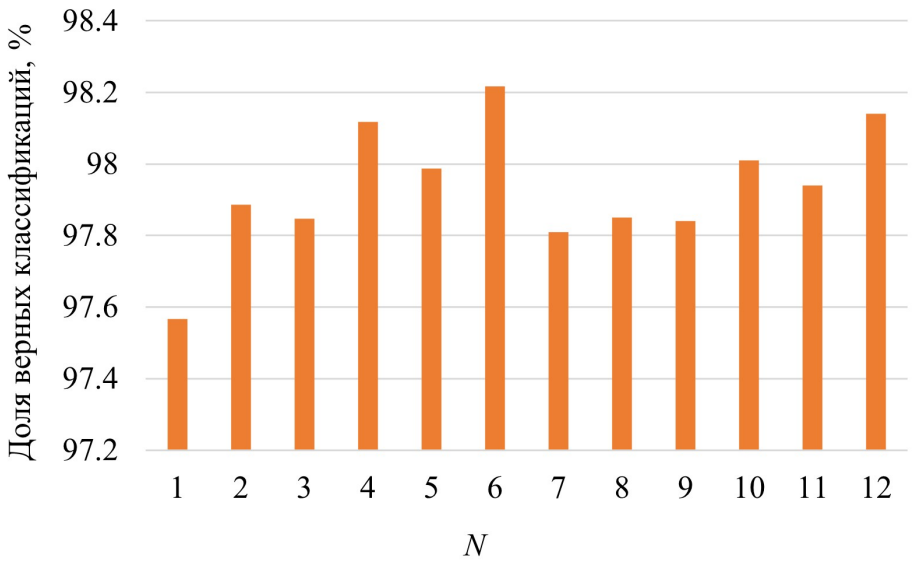


Рисунок 2.6 — Зависимость качества обучения СНС с дистилляцией от количества СНС учителей в коалиции, где N – размер коалиции

Для подтверждения суждений, связанных с (2.11), был проведен эксперимент с несколькими моделями СНС, которые так же используются в основном исследовании данной работы. Экспериментальные доказательства, связанные с дистилляцией СНС, будут представлены в следующем параграфе после обсуждения моделей СНС.

2.3.2 Разработка математических моделей сверточных нейронных сетей

Все рассматриваемые в исследовании СНС учителя работают с набором данных MNIST [127] и построены по общей архитектуре: сверточные, скрытые и выходные слои с функциями активации ReLU (алг. 2.1).

Сети ученики представлены в 5 видах. Это было вызвано следующими обстоятельствами. Разработчики TenSEAL продемонстрировали результат работы СНССК, в которой используется квадратичная функция активации. В предложенной сети ученика используется функция активации ELU (Exponential Linear Unit), однако в явном виде в ПГШ представить ее невозможно, поэтому она будет представлена в виде приближающего полинома.

Алгоритм 2.1. $ConvNet_{teacher}$

Вход: Изображение – $img(28 \times 28)$

Выход: Результат распознавания – out

1: Слой 1: $img \rightarrow Convolutional2D(1_{channels} \times 32_{filters})$,$(kernel\_size = 5, stride = 1, padding = 2), Maxpool(kernel = 2, stride = 2) \rightarrow x$
2: Функция активации: ReLU
3: Слой 2: $x \rightarrow Convolutional2D(32_{channels} \times 2_{filters}), kernel\_size = 5, stride = 1, padding = 2), Maxpool(kernel = 2, stride = 2) \rightarrow x$
4: Функция активации: ReLU
5: Слой 3: $x \rightarrow Linear(3136 \times 3136) \rightarrow x$
6: Функция активации: ReLU
7: Слой 4: $x \rightarrow Linear(3136 \times 3136) \rightarrow x$
8: Функция активации: ReLU
9: Слой 5: $x \rightarrow Linear(3136 \times 3136) \rightarrow x$
10: Функция активации: ReLU
11: Слой 6: $x \rightarrow Linear\,(3136 \times 10) \rightarrow out$
12: **Возвратить** out

В целом же, различия между открытыми моделями заключаются лишь в используемых функциях активации, поэтому рассмотрен общий алгоритм для СНС ученика (алг. 2.2).

Алгоритм 2.2. $ConvNet_{student}$

Вход: Изображение – $img(28 \times 28)$

Выход: Результат распознавания – out

1: Слой 1: $img \rightarrow Convolutional2D(1_{channels} \times 4_{filters})$, $(kernel\_size = 7, stride = 3, padding = 0) \rightarrow x$
2: Функция активации: $*$
3: Слой 2: $x \rightarrow Linear(256 \times 64) \rightarrow x$
4: Функция активации: $*$
5: Слой 3: $x \rightarrow Linear(64 \times 10) \rightarrow out$
6: **Возвратить** out
7: $\triangleright$ $*$ – Кортеж приближенных функций активации: Приближенная ELU (ПЕ), Квадратичная Функция Активации (КФА), Функция Активации с Обучаемыми Коэффициентами (ФАОК)

Таким образом, имеется следующий набор СНС:

1. $ConvNet_{teacher}$
2. $ConvNet_{student}$
3. $ConvNet_{student}$КФА, КФА
4. $ConvNet_{student}$КФА, ПЕ
5. $ConvNet_{student}$КФА, ФАОК
6. $ConvNet_{student}$ПЕ, ФАОК

Под ФАОК понимается оригинальная функция активации, суть которой в том, что во время обучения нейронной сети происходит модификация не только весов, но и коэффициентов полинома. Для полинома вида:

$$p(x) = x^n \cdot \alpha_1 + x^{n-1} \cdot \alpha_2 + \ldots + x^1 \alpha_{n-1} + \alpha_n, \tag{2.12}$$

где n – степень полинома, коэффициенты α_i подбираются с помощью градиентного спуска при статичных значениях весов. Таким образом можно повысить точность работы не только СНС, но и других моделей НС. Однако, в таком случае НС потребуется 2 прохода обратного распространения ошибки за одну итерацию обучения, что повышает затраты ресурсов на обучение НС. Рассмотрим теорему, доказывающую возможность применения такой функции активации.

Теорема 2. *Любую функцию $f(x)$, имеющую конечное число разрывов первого рода, можно аппроксимировать с помощью нейронной сети на ограниченной области с требуемой точностью, которая зависит от степени полинома аппроксимирующей функции $\hat{f}(x)$.*

Доказательство. Построим интерполяционный полином Лагранжа [7], выбрав узлы интерполяции таким образом, чтобы среди них оказались все точки разрыва первого рода. При таком построении разрывы первого рода устранятся, и полученная полиноминальная функция $\hat{f}(x)$ будет непрерывной.

Далее, согласно теореме об универсальной аппроксимации Цибенко [61], которая гласит, что непрерывная функция может быть аппроксимирована нейронной сетью при условии достаточного количества нейронов скрытого слоя с любой точностью, полиноминальная непрерывная функция $\hat{f}(x)$, построенная на базе функции с точками разрыва первого рода $f(x)$, может быть аппроксимирована нейронной сетью. □

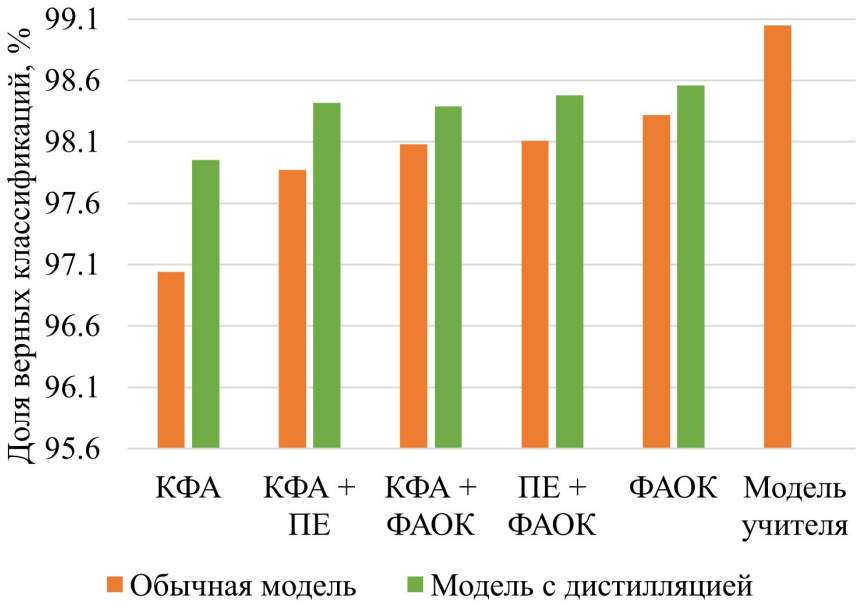


Рисунок 2.7 — Экспериментальное сравнение СНС, $a = 0.875$, размер коалиции учителей – 6

Теперь вернемся к выражению (2.11) и обратимся к Рисунку 2.7.

Под моделью учителя понимается сеть-учитель, под обычной СНС понимается СНС обученная без применения дистилляции. Как можно видеть на графике, модель СНС ученика на базе ФАОК показывает хороший результат и позволяет повысить точность классификации. При дистилляции СНС основанной на квадратичных функциях активации показывает самый большой прирост точности от эффекта дистилляции. Кроме того, результатом, характеризующим дистилляцию, так же является размер СНС, рассмотрим полученные результаты (табл. 4).

Таблица 4 — Размеры СНС

Модель	Размер, байт
Модель учителя	118,389,970
Обычная СНС	72,330
СНС только с КФА	72,310
СНС с КФА и ПЕ	72,280
СНС с КФА и ФАОК	73,568
СНС с ПЕ и ФАОК	73,259
СНС только сФАОК	72,264

Таким образом, удалось сжать сеть более чем в 1609 раз. Подробнее разница в размерах дистиллированных СНС рассмотрена на Рисунке 2.8.

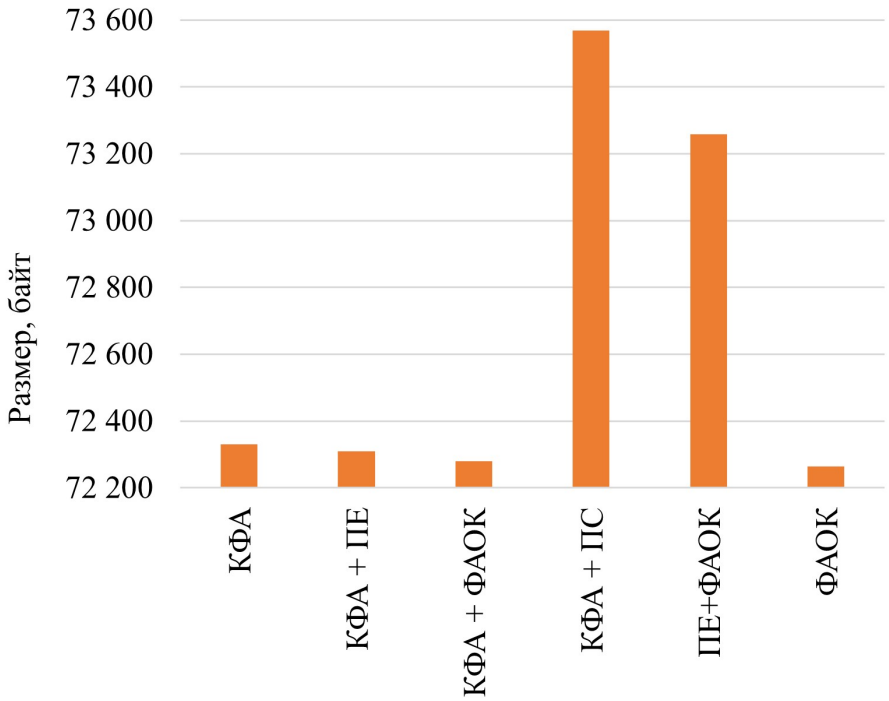


Рисунок 2.8 — Сравнение размеров СНС с дистилляцией

Больший размер СНС с ФАОК объясняется тем, что необходимо так же хранить и коэффициенты полиномов для функции активации ФАОК. В целом, если сравнивать с наиболее точной версией СНС – СНС с ФАОК, размер сети уменьшен более чем в 1627 раз при потере точности ± 0.5-1%. Очевидно, что в случае ФАОК точность зависит от степени полинома. Учитывая дальнейшее применение с ПГШ, желательно использовать малые степени полинома для удержания небольшой мультипликативной глубины. Для ФАОК коэффициенты подбираются в процессе обучения.

Поскольку разработчики TenSEAL уже показали результаты СНС на основе квадратичной функции активации в ПГШ, сосредоточимся на исследовании СНС с полиномиальными функциями активации.

2.3.3 Исследование математических моделей полиномиальных сверточных нейронных сетей сохраняющих конфиденциальность

Выбор параметров ПГШ для схемы CKKS является достаточно трудной задачей. Помимо того, что они отвечают за стойкость шифра и количество разрешенных операций, они оказывают влияние на вычислительную сложность операций и точность их результата. Так как в данном исследовании СНССК разрабатываются на базе библиотеки TenSEAL, то настройка параметров будет соответствовать парадигме Contex из SEAL.

Парадигма заключается в следующем. Пользователю предлагается задать степень модуля главного полинома CKKS, глобальный масштаб (количество знаков после запятой), а также размер коэффициентов в битах. Для СНС на основе КФА TenSEAL предлагает следующие параметры (табл. 5).

Таблица 5 — Параметры СНС с КФА

Степень модуля главного полинома	Размер коэффициентов модуля, бит	Глобальный масштаб, бит
8192	[31, 26, 26, 26, 26, 26, 26, 31]	26

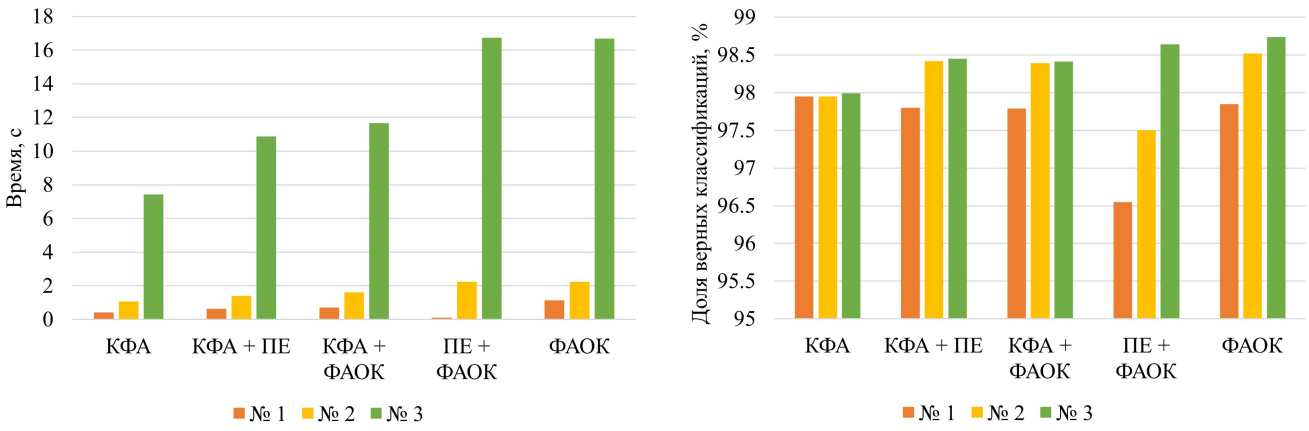
Однако данные параметры не подходят в случае использования функций активации ПЕ и ФАОК. Несмотря на то, что мультипликативной глубины теоретически достаточно для расчета КФА-ПЕ и КФА-ФАОК, точность операций будет не предсказуемой. Поэтому были протестированы следующие наборы параметров (табл. 6).

Если в случае №2 выбор коэффициентов был достаточно прост, то в случае №3 вопрос выбора модулей крайне сложен. SEAL хоть и поддерживает степень 32768, но не гарантирует безопасность при размере коэффициентов модулей более 59 бит. Поэтому для получения наиболее адекватного набора параметров было сделано следующее: 59 бит было выбрано в качестве начального значения, в случае успешного завершения работы СНС значение уменьшалось на 1 бит, и так до тех пор, пока операции выполнялись корректно. 53 бита в качестве краевых и 52 бита для центральных размеров являются наименьшими размерами,

Таблица 6 — Исследуемые наборы параметров для СНССК

Номер набора	Степень модуля главного полинома	Размер коэффициентов модуля, бит	Глобальный масштаб, бит
1.	8192	[31, 26, 26, 26, 26, 26, 26, 31]	26
2.	16384	[59, 40, 40, 40, 40, 40, 40, 59]	40
3.	32768	[53, 52, 52, 52, 52, 52, 52, 52, 52, 52, 52, 53]	52

при которых гомоморфные операции выполняются корректно. Для оценки качества и производительности полученной сети были выбраны точность и время, необходимое на один запуск СНССК (рис. 2.9).



а) б)

Рисунок 2.9 — Экспериментальное сравнение различных наборов параметров ПГШ для CKKS: а) по скорости работы СНССК; б) по точности СНССК

Исходя из полученных экспериментальных данных можно заметить следующую тенденцию. Вычислительная сложность шифра быстро растет с увеличением степени модуля главного полинома, на наборе №3 производительность снижается в несколько раз. При этом прирост точности составляет десятые доли процента для большинства моделей. Точность увеличивается на несколько процентов у моделей, которые содержат только полиномиальные

функции активации. Таким образом, можно сделать вывод, что модели CHCCK с ПЕ и ФАОК крайне чувствительны к выбору параметров шифрования схемы CKKS.

Полученные результаты позволяют говорить: дистилляция позволяет получить СНС по скорости обработки данных практически идентичную малым по размеру СНС без дистилляции, но при этом более точные за счет подхода к обучению с учителем в виде большей СНС. Кроме того, был оптимизирован процесс дистилляции для CHCCK. Исследование полиномиальных функции активации совместно со схемой CKKS показало высокую чувствительность к параметрам шифра, и быстрый рост вычислительной сложности при больших размерах модуля.

В контексте построения CHCCK интерес так же вызывают схемы BFV и TFHE. Разработчики библиотеки SEAL определяют BFV как наиболее точную схему ПГШ на параметрах с адекватной вычислительной сложностью. Однако, BFV имеет недостаток, связанный с накоплением внутренней ошибки, который необходимо исследовать. Кроме того, у BFV есть еще один существенный минус, данная схема работает с целыми числами. Преодолеть данный недостаток позволяет еще один способ оптимизации нейронных сетей. Квантизация нейронных сетей позволяет перевести, как значения весов, так и входные данные из вещественного типа в целый, обычно длиной 8 бит. Этот прием, как правило, используется для экономии оперативной памяти в больших моделях, но может быть эффективен и при работе с ПГШ. В следующем параграфе рассматриваются вопросы квантизации СНС и разработки CHCCK на базе других схем, помимо CKKS.

2.4 Квантизация параметров сверточной нейронной сети

Как у же было сказано выше, схемы BFV и TFHE не могут работать с вещественными числами. В СНС, как и в других НС, веса вещественные, а данные зачастую нормализуются в диапазон $[0, 1]$, что создает проблемы для схем BFV и TFHE. Для того, чтобы решить данные проблемы необходимо прибегнуть к квантизации.

Для обработки вещественных чисел в целочисленной арифметике необходимо привести их к целочисленному виду. Идея квантизации чисел взята из квантизации сигнала [142], подразумевающей приведение аналогового сигнала в дискретный вид для дальнейшей цифровой обработки. Как и в случае сигнальной квантизации, существует несколько подходов к числовой квантизации, из которых можно выделить три основных: квантизация методом 2^N, аффинная квантизация и симметричная квантизация. Рассмотрим каждый из них подробнее.

2.4.1 Квантизация методом 2^N

Квантизация методом 2^N предполагает повышение разрядности рационального числа, при котором значимые цифры мантиссы переходят в разряды чисел перед запятой. Степень двойки используется для замены операции умножения гораздо более эффективной операцией битового сдвига. Рассматриваемый метод квантизации можно описать Алгоритмом 2.3.

Алгоритм 2.3. Квантизация методом 2^N

Вход: x_{float}, N

Выход: x_{int}

1: $x_{int} = round(x_{float} \cdot 2^N)$

2: **Возвратить** x_{int}

Преимуществом данного метода является простота: для квантизации (деквантизации) необходимо умножить (разделить) число на 2^N. Недостаток данного метода заключается в том, что в зависимости от требований к точности вычислений, N может достигать больших значений, для хранения которых может понадобиться больше памяти. Однако, несмотря на недостатки, данный метод широко применяется в области целочисленных вычислений. С точки зрения применения в СНС, метод менее популярен по сравнению с другими ввиду того, что требуются большие значения степени N.

2.4.2 Аффинная квантизация

Данный вид квантизации основан на аффинных преобразованиях [167]. Основным отличием данного метода является то, что ноль смещается на числовой прямой. Для проведения квантизации данного вида необходимо: определить нулевую точку и масштаб. Они определяются следующими выражениями

$$S = \frac{x_{max} - x_{min}}{q_{max} - q_{min}},$$

где S – масштаб, x_{max}, x_{min} – максимальное и минимальное значения множества чисел соответственно, q_{max}, q_{min} – верхняя и нижняя границы квантизации соответственно. В контексте обработки чисел, значения q выбираются в зависимости от степени двойки, соответствующей желаемой длине чисел. То есть если предполагается, что числа будут длиной 8 бит, то исходя из выражения $2^8 = 256$, верхней границей будет 127 (первое значение зарезервировано для нуля), а нижней -128. Нулевая точка определяется выражением

$$Z = \left\lfloor q_{min} - \frac{x_{min}}{S} \right\rceil,$$

где Z – нулевая точка, $\lfloor \ \rceil$ – операция округления к ближайшему целому. Сама квантизация определяется следующей формулой

$$x_q = \left\lfloor \frac{x}{S} + Z \right\rceil.$$

Аффинная квантизация (алг. 2.4), за счет жесткого задания нулевой точки, подходит для несимметричных распределений, что позволяет эффективно квантовать нейронные сети использующие в качестве функции активации ReLU.

2.4.3 Симметричная квантизация

В отличие от аффинной квантизации – симметричная предполагает, что ноль не смещен и имеет значение 0. По-сути это упрощенная аффинная квантизация, где $Z = 0$ [106], а для задания масштаба квантизации, как и в случае 2^N,

Алгоритм 2.4. Аффинная квантизация

Вход: x_{range}, N, x_{float}

Выход: x_{int}

1: $x_{max} = max(x_{range})$
2: $x_{min} = min(x_{range})$
3: $q_{max} = 2^{N-1} - 1$
4: $q_{min} = -2^{N-1}$
5: $S = \frac{x_{max} - x_{min}}{q_{max} - q_{min}}$
6: $Z = round(q_{min} - \frac{x_{min}}{S})$
7: $x_{int} = round(\frac{x_{float}}{S} + Z)$
8: **Возвратить** x_{int}

в качестве границ значений применяется степень двойки. Однако, для достижения симметричности необходимо пожертвовать еще одним значением, т.е. если $N = 8$, то в случае аффинной квантизации диапазон $[-128, 127]$, а в случае симметричной $[-127, 127]$. Тогда масштаб S определяется выражением

$$S = \frac{x_{max} - x_{min}}{2^N - 2}.$$

Сама же квантизация определяется следующей формулой

$$x_q = \left\lfloor \frac{x - x_{min}}{S} - 2^{N-1} + 1 \right\rceil.$$

С одной стороны, симметричная квантизация (алг. 2.5) имеет то же преимущество, что и 2^N, требует меньше памяти для хранения констант, с другой стороны, она менее точна для асимметричных распределений.

Таким образом, выбор метода квантизации напрямую зависит от модели СНС, требований к точности и ограничений на адресное пространство памяти.

2.4.4 Сравнительный анализ методов квантизации

Проанализируем результаты экспериментального исследования рассмотренных видов квантизации чисел на примере СНС. Сеть была предварительно обучена, после чего квантована. Эксперимент был организован следующим образом. Пул из 50 нейронных сетей обучался в формате float32. После этого

Алгоритм 2.5. Симметричная квантизация

Вход: x_{range}, N, x_{float}

Выход: x_{int}

1: $x_{max} = max(x_{range})$

2: $x_{max} = min(x_{range})$

3: $S = \frac{x_{max} - x_{min}}{2^N - 2}$

4: $x_{int} = round(\frac{x_{float} - x_{min}}{S} - 2^N + 1)$

5: **Возвратить** x_{int}

каждая из 50 сетей квантовалась каждым из рассмотренных методов в int4, int8 и int16. Далее для каждой сети на вход подавался тестовый набор данных, для обучения и тестирования использовался набор данных MNIST. На основе полученных данных определялись средние величины измеряемых характеристик для каждого из методов квантизации сетей. В качестве измеряемых характеристик были выбраны: производительность (время, затраченное на один тест), точность ответа нейронной сети и потребляемая память. Рассмотрим полученные результаты.

В первую очередь рассмотрим производительность (рис. 2.10).

Как и ожидалось, квантованные модели СНС работают быстрее. Наилучшей производительностью обладают модели, квантованные методом 2^N при длине адресного пространства 4 бита. Однако, производительность является не единственным показателем, который представляет интерес в данном исследовании. Далее рассмотрим точность (рис. 2.11).

СНС квантованные методом 2^N при длине адресного пространства 8 бит показывают достаточно высокий результат по точности, практически равный изначальному. Однако, в данном случае за точность приходится расплачиваться занимаемой памятью (рис. 2.12).

За счет того, что данные умножаются на 2^N – число получается достаточно длинным, кроме того, для данного метода квантизации подписи int4, int8, int16 достаточно условные, так как здесь имеется в виду именно N. Полученные результаты можно считать открытыми к обсуждению и несущими скорее информативный характер. В зависимости от решаемой задачи, а также имеющейся аппаратной базы, необходимо использовать определенный тип квантизации. Например, в случае со схемой BFV, где длина зашифрованного числа зависит от параметра шифра, потребляемой квантованными значениями

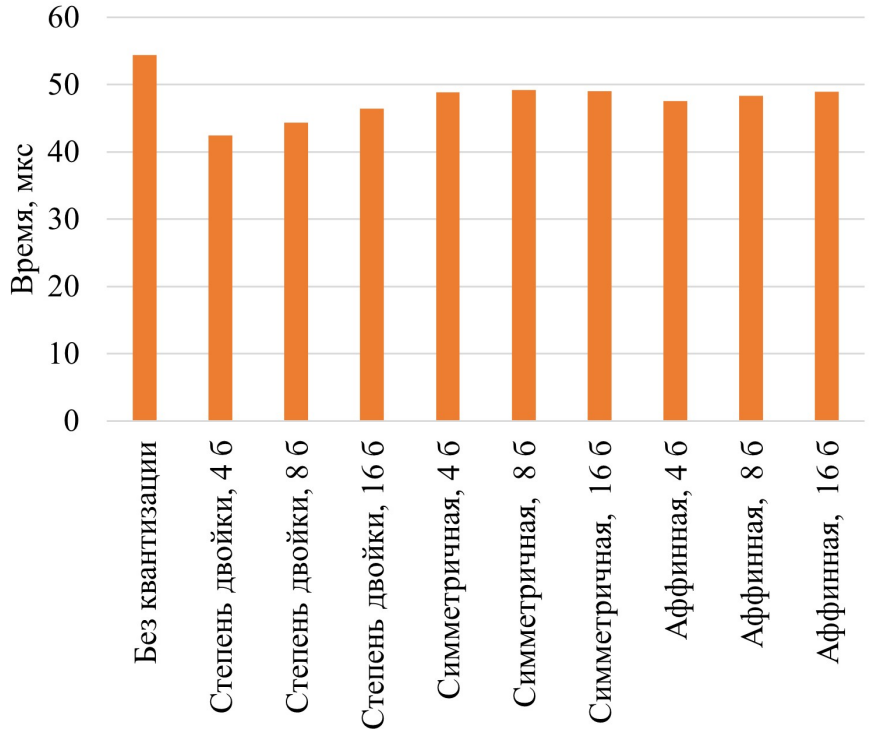


Рисунок 2.10 — Производительность СНС, квантованных различными методами

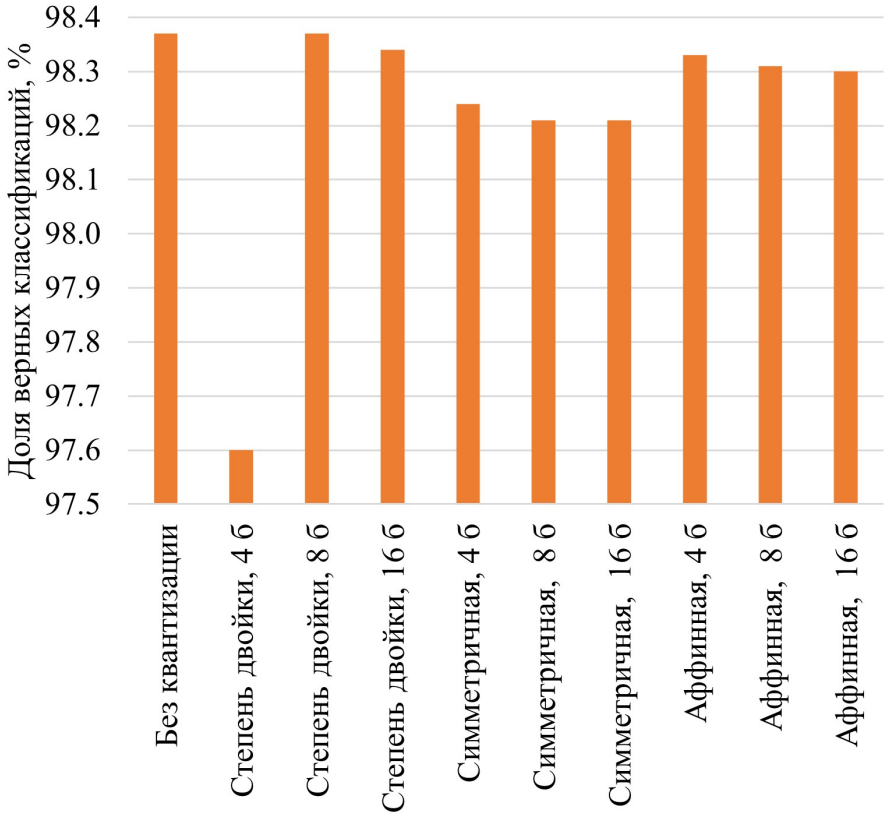


Рисунок 2.11 — Точность СНС, квантованных различными методами

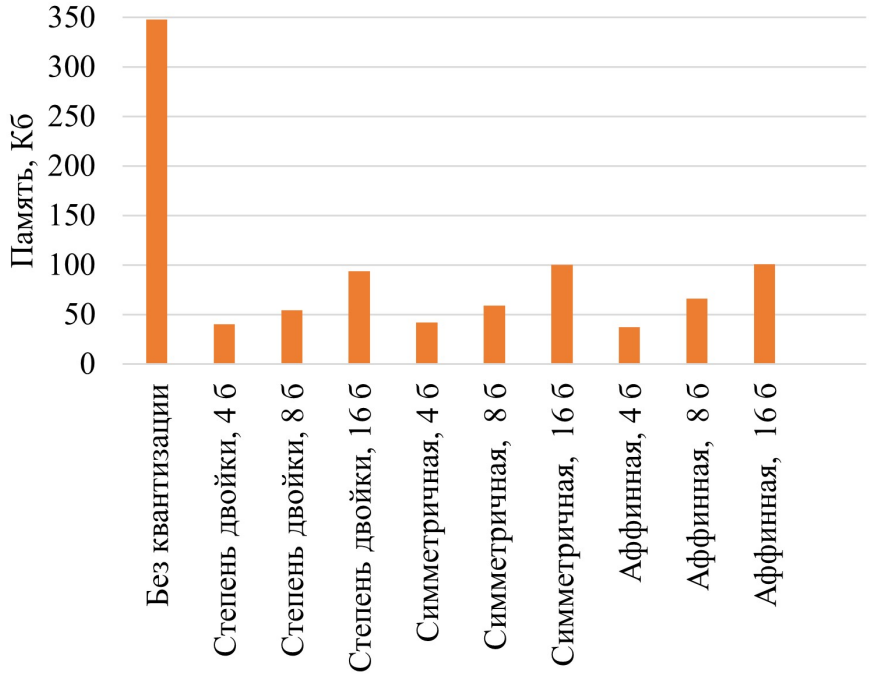


Рисунок 2.12 — Память, потребляемая СНС, квантованными различными методами

памятью можно пренебречь. В таком случае наиболее эффективным способом квантизации является метод 2^N.

2.4.5 Построение сверточной нейронной сети со схемой BFV

Если в случае схемы CKKS, которой можно шифровать и обрабатывать числа с фиксированной запятой, достаточно зашифровать данные и оптимизировать сеть, исключив операции деления и сравнения (использовать приближенные функции активации, либо квадратичную), то в случае BFV СНССК необходимо оптимизировать к целочисленному формату. Для этого в первую очередь необходимо квантовать СНС. Если применяются приближенные функции активации, коэффициенты полиномов необходимо преобразовать согласно выбранному методу квантизации. Для реализации операции свертки так же существуют алгоритмы, например, описанные в работе [74]. Таким образом, используя вышеперечисленные методы и алгоритмы, необходимо по-

строить приближенную модель СНС, пригодную для шифрования схемой ПГШ BFV.

В работе [74] предложена модель СНС со схемой BFV. В целом достигнут достаточно хороший результат, как пример того, на что способна схема BFV. В данном исследовании представлена оптимизация этой модели, полученная квантованием нейронной сети более быстрым способом, с сохранением при этом достаточно высокой точности, кроме того, квадратичные функции активации заменены на приближенные ReLU в квантованном виде. В случае ПГШ со схемой BFV Maxpooling заменяется на Averagepooling. Вычисление Maxpooling предполагает поиск максимального элемента, что в схемах ПГШ в настоящий момент невозможно, либо крайне затратно (в случае схемы CKKS). Выполнить Averagepooling в схеме BFV возможно путем суммирования шифртекстов и умножения на константу, известную заранее, зависящую от параметров ядра свертки, ее можно либо предварительно зашифровать, либо хранить в открытом виде. В данном исследовании предлагается хранить ее в открытом виде, так как во время обучения СНС она хранится в открытом виде, а, следовательно, известна и ее шифрование не имеет смысла.

Рассмотрим приближение функции активации ReLU и ее квантизацию.

Приблизим ReLU с помощью полиномов Чебышева первого рода второй степени. Получим следующий результат.

$$\hat{y}(x) = 8.123 \cdot 10^{-2}x^2 + 5 \cdot 10^{-1}x + \ 5.918 \cdot 10^{-1}. \tag{2.13}$$

При этом коэффициент детерминации $R^2 = 0.987$. График данного приближения представлен на Рисунке 2.13.

При этом квантованная ReLU и ее приближение (2.13) выглядят следующим образом (рис. 2.14).

На основе данной функции будет построена функция активации СНССК.

Прямой проход по СНССК организован на основе [96]. Так как весовые коэффициенты СНССК получены при обучении в открытом виде, их шифрование нецелесообразно, что так же позволяет сэкономить вычислительные ресурсы.

Рассмотрим полученную СНССК. Для этого нужно рассмотреть вспомогательные алгоритмы, а именно инициализацию шифрования, сверточный слой, функцию активации, линейный слой. Для начала создается структура контекста, которая определяется на основе параметров шифрования (алг. 2.6).

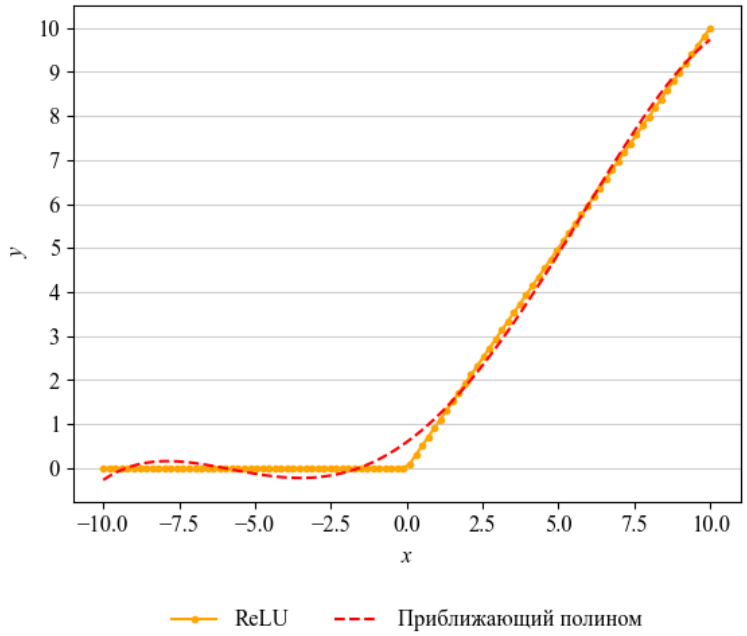


Рисунок 2.13 — Приближение функции активации ReLU

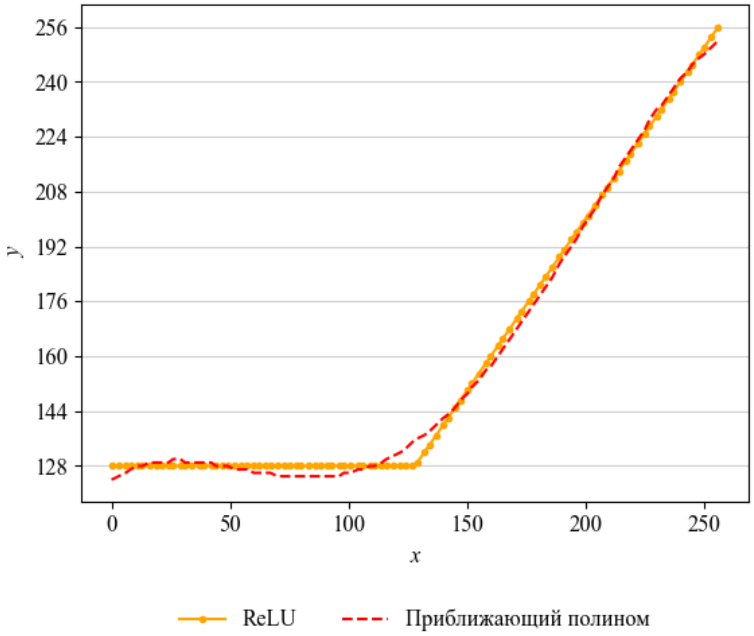


Рисунок 2.14 — Приближение квантованной функции активации ReLU

Далее рассмотрим функцию шифрования изображения (алг. 2.7). Шифрование в ПГШ производится векторами.

Операция свертки требует дополнительных переменных для хранения векторов весов и смещений, шага, количества каналов, весов фильтров и размера ядра свертки. Так же потребуются несколько гомоморфных операций, а именно сложение и умножение с открытым текстом ($\text{Add}_{\text{Plain}}$, $\text{Mult}_{\text{Plain}}$), сложение

Алгоритм 2.6. Формирование параметров шифрования (функция *Encryption_Context*)

Вход: $\Theta(n, p, q)$

Выход: *context*

1: $context \leftarrow (k_p, k_q), \Phi_{n,q}$

2: **Возвратить** *context*

Алгоритм 2.7. Шифрование изображения (функция *Encryption_Image*)

Вход: *image*, *context*

Выход: $image_{enc}$

1: $m \leftarrow length(image)$

2: **Цикл от** $i = 0$ **до** $m - 1$ **выполнять**

3: $\quad image_{enc,i} \leftarrow context(image_i)$

4: **Конец цикла**

5: **Возвратить** $image_{enc}$

двух шифртекстов (Add), а также операция вращения (Rot). Предполагается, что размер изображения M, N известен (алг. 2.8).

Теперь рассмотрим алгоритм прямого распространения, где P – количество нейронов предыдущего слоя, Q – число нейронов следующего слоя (алг. 2.9).

Алгоритм применения функции активации достаточно прост. Необходимо получить зашифрованный полином (алг. 2.10).

Наконец, реализующий СНССК алгоритм выглядит следующим образом (алг. 2.11)

Таким образом, на основе представленных алгоритмов была реализована СНССК. Для шифрования использовалась библиотека SEAL, для обучения СНС в открытом виде библиотека PyTorch.

Алгоритм 2.8. Конфиденциальный сверточный слой CHCCK (функция $Layer_{conv_{enc}}$)

Вход: $image_{enc}$, $kernel\_size$, $stride$, w_{filter}, b_{filter}, $context$

Выход: $image_{enc\_conv}$

1: M, $N \leftarrow size(image_{enc})$
2: **Цикл от** $i = 0$ **до** $M - 1$ **выполнять**
3: **Цикл от** $j = 0$ **до** $N - 1$ **выполнять**
4: $Temp_{mult_j} \leftarrow Mult_{Plain}(image_{enc_j}, w_{filter_i}, context)$
5: **Конец цикла**
6: **Конец цикла**
7: **Цикл от** $i = 0$ **до** $M - 1$ **выполнять**
8: **Цикл от** $j = 0$ **до** $\log_2 kernel\_size - 1$ **выполнять**
9: $Temp_{rot_i} \leftarrow Rot(Temp_{mult_i}, j, context)$
10: $Temp_{add_i} \leftarrow Add(Temp_{rot_i}, Temp_{mult_i}, context)$
11: $image_{enc_{conv},i} \leftarrow Add_{Plain}(Temp_{add_i}, kernel_{size}, context)$
12: **Конец цикла**
13: **Конец цикла**
14: **Цикл от** $i = 0$ **до** $N - 1$ **выполнять**
15: $image_{enc_{conv}} \leftarrow Add_{Plain}(image_{enc_{conv},i}, b_{filter,i}, context)$
16: **Конец цикла**
17: **Возвратить** $image_{enc_{conv}}$

2.4.6 Исследование разработанной модели CHCCK

Выбор оптимальной CHCCK сводится в рамках данного исследования к сравнению следующих моделей: CHCCK с CKKS, CHCCK с BFV из работы [74] и CHCCK с BFV предложенной в этом исследовании. Стоит отметить, что в реализации из работы [74] использовалась библиотека Pyfhel. Если рассмотреть данную библиотеку, то она основана на SEAL, и по-сути создает интерфейс взаимодействия с ней на языке Python. Рассмотрим полученные результаты (рис. 2.15).

Полученные результаты можно интерпретировать следующим образом. В работе [74] авторы нетривиальным образом реализуют шифрование весов CHCCK в BFV. Если проанализировать программный код, находящийся в

Алгоритм 2.9. Конфиденциальный прямой проход CHCCK (функция *Encryption_Forward_Propagation*)

Вход: $image$, w, b, $context$

Выход: $image_{enc\_forward}$

P, $Q \leftarrow w$
$Temp = 0$
$Temp_{rot} = 0$
Цикл от $i = 0$ **до** $P - 1$ **выполнять**
    $Temp_{rot_i} \leftarrow Rot(image,\ \log_2 Q,\ context)$
    Цикл от $j = 0$ **до** $Q - 1$ **выполнять**
        $Temp_i \leftarrow Mult_{Plain}\,(Temp_{rot_i},\ w_j,\ context)$
    Конец цикла
Конец цикла
Цикл от $i = 0$ **до** $Q - 1$ **выполнять**
    $image_{enc\_forward_i} \leftarrow Add_{Plain}(Temp_i,\ b_i,\ context)$
Конец цикла
Возвратить $image_{enc\_forward}$

Алгоритм 2.10. Конфиденциальная ПР (функция *Encryption_Approximate_ReLU*)

Вход: x, $coefficients$, $contex$

Выход: x_{act}

$temp = 1$
Цикл от $i = 0$ **до** $length(coefficients) - 1$ **выполнять**
    Цикл от $j = 0$ **до** $length(coefficients) - i - 1$ **выполнять**
        $temp \leftarrow Mult(temp,\ x,\ context)$
    Конец цикла
    $x_{act} \leftarrow Add(x,\ Mult_{Plain}(temp,\ coefficients_i,\ context),\ context)$
Конец цикла
Возвратить x_{act}

Алгоритм 2.11. CHCCK

Вход: $image$, n, p, q, $kernel\_size$, $stride$, w_{filter}, b_{filter}, w_{hidden}, b_{hidden}, w_{output}, b_{output}, $coefficients$

Выход: $answer$

1: $context \leftarrow Encryption\_Context(n, p, q)$

2: $x_{enc} \leftarrow Encryption\_Image\,(context) \leftarrow image$

3: $x_{enc} \leftarrow Layer_{conv_{enc}}(x_{enc}, kernel\_size, stride, w_{filter}, b_{filter}, context)$

4: $x_{enc} \leftarrow Encryption\_Approximate\_ReLU(\ coefficients, contex) \leftarrow x_{enc}$

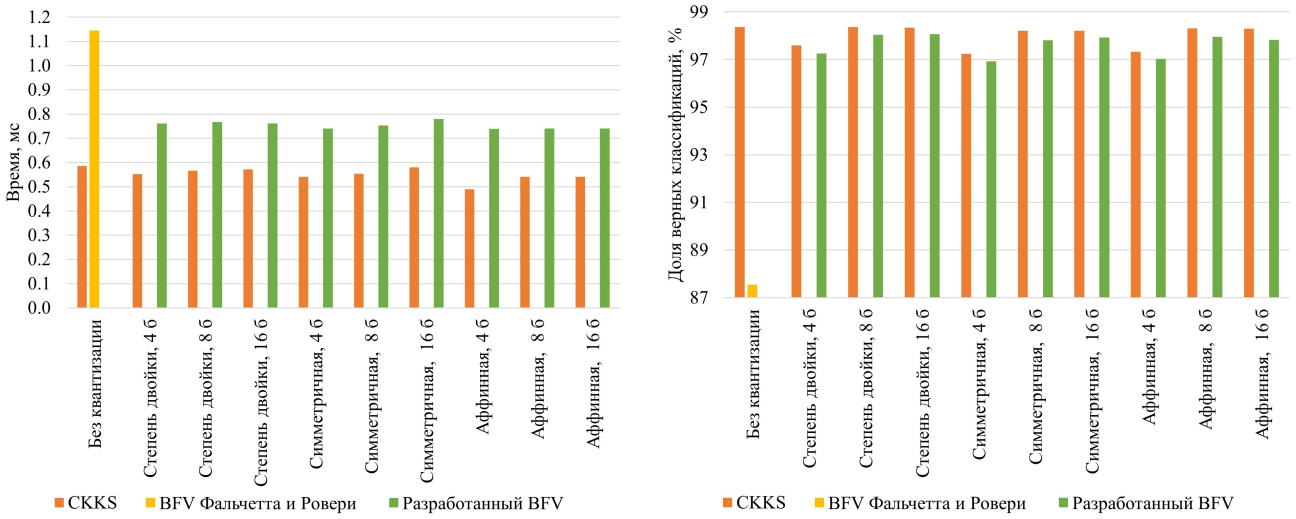
5: $x_{enc} \leftarrow Encryption\_Forward\_Propagation(x_{enc}, w_{hidden}, b_{hidden}, context)$

6: $x_{enc} \leftarrow Encryption\_Approximate\_ReLU(\ coefficients, contex) \leftarrow x_{enc}$

7: $x_{enc} \leftarrow Encryption\_Forward\_Propagation(x_{enc}, w_{output}, b_{output}, context)$

8: $answer \leftarrow Decription \leftarrow x_{enc}$

9: **Возвратить** $answer$



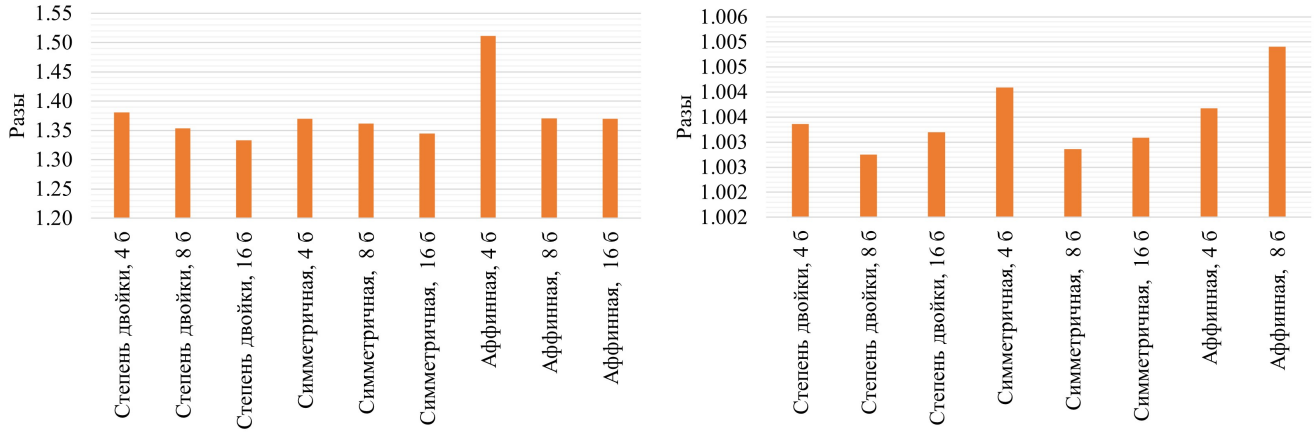
а) б)

Рисунок 2.15 — Экспериментальное сравнение квантованных моделей CHCCK: а) по времени работы; б) по точности распознавания

открытом доступе, авторы с помощью вычислительных механизмов шифруют число, представленное в дробном виде, что, конечно, упрощает работу со CHCCK, основанными на схеме BFV, но сказывается на их производительности и точности. Кроме того, авторы не задавались целью разработать наиболее эффективную модель CHCCK, их исследование достаточно важно, как демонстрация возможности создания CHCCK с BFV. Целью добавления данной работы в сравнительный анализ было наглядно показать развитие методов ПГШ и CHCCK в целом.

В случае CKKS можно заметить (рис. 2.15), что квантизация значений оказывает влияние на скорость обработки данных, кроме того, целочисленные значение меньшей длины позволяют применять менее вычислительно сложные параметры. Точность классификации так же зависит от того, каким методом и до какой длины были квантизированны веса.

В случае разработанного BFV, с понижением разрядности квантизации ожидаемо снижается точность модели (рис. 2.15б)). Как можно заметить на Рисунке 2.15а), способ квантизации влияет не только на точность, но и на скорость обработки. Предложенная CHCCK со схемой BFV проигрывает по скорости модели со схемой CKKS, аффинная квантизация показывает наилучший результат по скорости. Рассмотрим отношения показателей квантованных моделей со схемой CKKS к аналогичным показателям моделей со схемой BFV (рис. 2.16).



а) б)

Рисунок 2.16 — Отношения показателей квантованных CHCCK со схемой CKKS к CHCCK со схемой BFV: а) времени работы; б) точности распознавания

На Рисунке 2.16 более отчетливо просматривается, приемущество схемы CKKS, как по времени работы, так и по точности распознавания. Однако, данный результат не сравним с CKKS без квантизации. Подобные результаты можно объяснить тем, что в BFV внутренний рост шума увеличивается, как при приближении к лимиту операций, так и после бутстраппинга, что заметили авторы работы [74]. Для повышения точности требуется использовать параметры, которые значительно повышают вычислительную сложность. Квантизация хоть и улучшает ситуацию в целом, однако не дает BFV в полной мере конкурировать с CKKS в области CHCCK.

Помимо CKKS и BFV, как уже было сказано в Параграфе 2.1, разработана реализация схемы TFHE для исследования ИИ – Concrete-ML. Таким

образом, аналитический обзор сохраняющих конфиденциальность методов ИИ будет неполным без исследования применимости схемы TFHE. Стоит отметить тот факт, что TFHE и CKKS устроены совершенно по разному, как их системы шифрования, так и организация гомоморфной арифметики, что в теории позволит получить большее количество математических моделей СНССК для более широкого спектра задач.

2.5 Заключение по второй главе

В данной главе представлено исследование различных методов и алгоритмов оптимизации и повышения эффективности СНССК, проанализированы библиотеки программной реализации методов ПГШ. Подробно рассмотрены наиболее релевантные для СНССК методы и алгоритмы повышения эффективности:

1. Модификация матричного умножения.
2. Дистилляция сверточных нейронных сетей.
3. Квантизация параметров сверточных нейронных сетей.

Модификация матричного умножения основана на факте, заключающемся в том, что веса СНС являются общедоступными, так как обучение СНС проводится в открытом виде. Шифрование весов не имеет смысла, так как они уже раскрыты. Таким образом, умножая зашифрованную матрицу на открытую, можно получить выигрыш как в скорости обработки данных, так и в потреблении памяти, за счет игнорирования конфиденциальности параметров СНС.

Дистилляция позволяет обучить нейронную сеть с участием ансамбля учителей, которыми выступают более крупные СНС, обученные на таком же наборе данных. Несмотря на большие затраты ресурсов на подготовительном этапе в виде обучения нескольких больших нейронных сетей, СНС получает более высокую точность при сравнительно малом размере, что позволяет применять ее более эффективно в связке с ПГШ.

Квантизация позволяет применять для СНС схемы ПГШ, которые не поддерживают обработку вещественных чисел. Примерами таких схем являются BFV и TFHE. В данной главе рассмотрена схема BFV. Полученные результаты

позволили улучшить как скорость обработки данных, так и точность CHCCK, однако точности CHCCK со схемой CKKS в вещественном виде все равно достичь не удалось, кроме того, схема BFV так же проигрывает и в скорости обработки данных.

Схема TFHE выглядит более перспективной по скорости обработки данных чем BFV ввиду наличия операций с LUT. В следующей главе представлены исследования, связанные с TFHE: разработка математических моделей CHCCK со схемой TFHE на базе рассмотренных в данной главе методов и алгоритмов, а так же их экспериментальное сравнение с аналогичными математическими моделями CHCCK со схемой CKKS. Полученные результаты позволят получить более целостную картину о возможностях применения CHCCK.

Глава 3. Разработка математических моделей нейронных сетей сохраняющих конфиденциальность

3.1 Математические модели нейронных сетей использующиеся в исследовании

В Параграфах 2.2 и 2.3 были разработаны и представлены алгоритмы для сверточных нейронных сетей сохраняющих конфиденциальность. В данном параграфе они рассмотрены более подробно в контексте использования в СНССК.

Как уже было отмечено в Параграфе 2.3, для повышения эффективности СНССК на базе ПГШ необходимо применение дистилляции, уменьшающей размеры и вычислительную сложность СНССК. Там же отражены результаты, которые показывают уменьшение потребления памяти и повышение скорости обработки (по сравнению с нейронными сетями большого размера) и повышение точности (по сравнению с нейронными сетями такого же размера) СНССК подвергнутых дистилляции.

В данном параграфе будут рассмотрены модели СНС, используемые в исследовании. В первую очередь будут рассмотрены модели СНС учителей, необходимые для дистилляции; далее модели СНС, использующие приближенные функции активации на основе полиномов, затем на основе квадратичной функции активации. Данные модели вынесены в отдельный параграф, так как они будут использоваться при разработке математических моделей как на основе CKKS, так и на основе TFHE. Кроме того, для исследований предполагается использовать 2 набора данных:

1. Набор рукописных цифр MNIST, с размером изображений 28×28;
2. Сжатый набор рукописных цифр, с размером изображений 8×8.

Это необходимо, потому что схемы CKKS и TFHE по разному обрабатывают данные. Ввиду особенностей накопления шума в схеме TFHE, существует мнение, что данная схема хуже обрабатывает длинные вектора чем малые. Использование двух наборов данных позволит получить результаты зависимости эффективности схем от размера обрабатываемых данных.

Главное отличие математических моделей СНС в таком случае будет заключаться в количестве входов.

3.1.1 Математические модели сверточных нейронных сетей учителей

Рассматриваемые модели СНС-учителей обладают следующими общими характеристиками:

- Количество сверточных слоев: 2;
- Размер ядер сверточных слоев: 5;
- Шаг свертки: 1 (и по ширине, и по высоте изображения);
- Паддинг: 2 (и по ширине, и по высоте изображения);
- Пуллинг: по максимальному элементу.

Первой рассмотрим математическую модель СНС-учителя, обучаемую на наборе данных MNIST с размером изображений 28×28. Обращаясь к Параграфу 1.1.2, построим первый сверточный слой для СНС-учителя на основе изображений в градациях серого (одноканальных) размером 28×28 с объемом партии ($batch\_size$) равным 64. Размер слоев представлен в следующем виде, сверточные слои $\{batch\_size, channels, M, N\}$, слои прямого распространения $\{P, Q\}$. В таком случае $M = 28$, $N = 28$, MNIST предоставляет изображения в градациях серого, следовательно, $channels = 1$. Таким образом, размеры сверточного слоя имеют следующий вид $\{64, 1, 28, 28\}$ на входе. Такая расстановка размеров наиболее оптимальна, т.к. при программной разработке СНС в первую очередь обрабатывается размер партии, далее количество каналов и наконец размер изображения. Размеры слоя после применения фильтров зависят, как от размеров сверточного слоя, так и от ядра свертки, в данном случае ядро сверти $k = 5$, количество фильтров – 32, следовательно учитывая паддинг и пулинг по максимальному элементу равный двум, получим размер выхода равный $\{64, 32, 14, 14\}$. Второй сверточный слой уже имеет размер выхода $\{64, 64, 7, 7\}$. После этого необходимо линеаризовать размерности тензоров до двумерных, чтобы выполнить прямой проход по скрытым слоям. Таким образом, получаем размер $\{64, 3136\}$. Следующие три слоя рассчитываются по формуле (1.1) и их размеры $\{3136, 3136\}$, размер выходного слоя $\{3136, 10\}$. Общий вид СНС-учителя для изображений размером 28×28 представлен на Рисунке 3.1.

Теперь рассмотрим реализацию данной СНС. Отдельно выделим алгоритм свертки, так как его применение аналогично во всех последующих рассматриваемых СНС. Кроме того, стоит рассмотреть здесь алгоритм пулин-

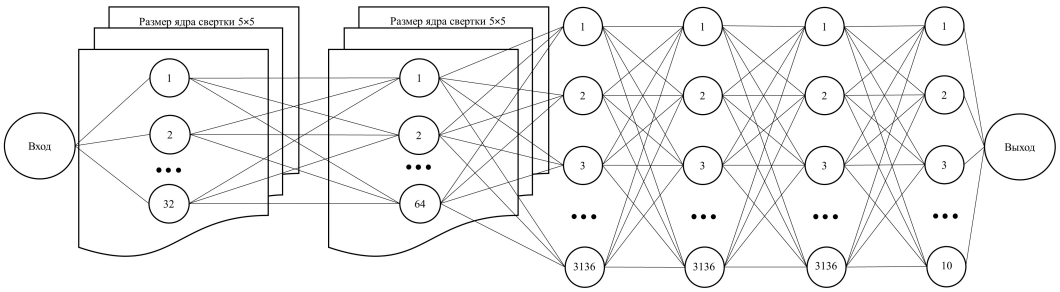


Рисунок 3.1 — Модель СНС-учителя для изображений размером 28×28

га по максимальному элементу, так как в компактных СНС он использоваться не будет (агл. 3.1).

Алгоритм 3.1. Пулинг по максимальному элементу (функция Max_pool)

Вход: $input\_tensor$, $pool\_size$

Выход: $output$

1: n, h, $w \leftarrow$ размеры $input\_tensor$
2: **Цикл от** $i = 0$ **до** $h/pool\_size - 1$ **выполнять**
3: **Цикл от** $j = 0$ **до** $w/pool\_size - 1$ **выполнять**
4: $region \leftarrow input\_tensor[:, i \cdot pool\_size : (i+1) \cdot pool\_size - 1, j \cdot pool\_size : (j+1) \cdot pool\_size - 1]$
5: $output[:, i, j] \leftarrow max(region)$
6: **Конец цикла**
7: **Конец цикла**
8: **Возвратить** $output$

Теперь, имея алгоритм пулинга по максимальному элементу, можно рассмотреть реализацию сверточного слоя (алг. 3.2).

В случае обучения СНС-учителя, изначально тензоры $wieghts$ и $bias$ заполняются случайными числами в диапазоне [0,1]. Так же рассмотрим реализацию слоя прямого распространения в общем виде (алг. 3.3).

Реализация СНС-учителя для изображений размера 28×28 целиком представлена Алгоритмом 3.4.

На основе данной математической модели было построено и обучено 6 СНС-учителей. Количество сетей учителей было выбрано исходя из результатов исследований, представленных в Параграфе 2.3. Аналогичным образом построим математическую модель СНС-учителя для изображений размера 8×8.

Построим первый сверточный слой для СНС-учителя, на основе изображений в градациях серого (одноканальных) размером 8×8, с объемом партии

Алгоритм 3.2. Сверточный слой (функция $Conv\_layer$)

Вход: $input\_tensor$, $weights$, $bias$, $stride$, $padding$

Выход: $output$

n, h, $w \leftarrow$ размеры $input\_tensor$
m, $k \leftarrow$ размеры $weights$
$padded\_input \leftarrow (input\_tensor, padding)$
$h\_p, w\_p \leftarrow$ размеры $(padded\_input)$
Цикл от $i = padding$ **до** $h\_p - padding - 1$ **шаг** $stride$ **выполнять**
    Цикл от $j = padding$ **до** $w\_p - padding - 1$ **шаг** $stride$ **выполнять**
        $region \leftarrow padded\_input[: , i - [\frac{k}{2}] : i + [\frac{k}{2}], j - [\frac{k}{2}] : j + [\frac{k}{2}]]$
        Цикл от $f = 0$ **до** $m - 1$ **выполнять**
            $output[f, i/stride, j/stride] \leftarrow sum(region \cdot weights[f]) + bias[f]$
        Конец цикла
    Конец цикла
Конец цикла
Возвратить $output$

Алгоритм 3.3. Слой прямого распространения (функция $Layer$)

Вход: $input\_tensor$, $weights$, $bias$, n, Q, P

Выход: $output$

$temp = 0$
Цикл от $i = 0$ **до** $n - 1$ **выполнять**
    Цикл от $j = 0$ **до** $Q - 1$ **выполнять**
        Цикл от $k = 0$ **до** $P - 1$ **выполнять**
            $temp[i][j] \leftarrow temp[i][j] + input\_tensor[i][k] \cdot weights[k][j]$
        Конец цикла
        $output[i][j] \leftarrow temp[i][j] + bias[j]$
    Конец цикла
Конец цикла
Возвратить $output$

Алгоритм 3.4. СНС-учитель для изображений 28×28

Вход: $input\_tensor(64,\ 28,\ 28)$, $k = 5$, $weights_{conv_1}(1,\ 32,\ k,\ k)$, $bias_{conv_1}(32)$, $weights_{conv_2}(32,\ 64,\ k,\ k)$, $bias_{conv_2}(64)$, $weights_{layer_1}(3136,\ 3136)$, $bias_{layer_1}(3136)$, $weights_{layer_2}(3136,\ 3136)$, $bias_{layer_2}(3136)$, $weights_{layer_3}(3136,\ 3136)$, $bias_{layer_3}(3136)$, $weights_{layer_4}(3136,\ 10)$, $bias_{layer_4}(10)$, $stride = 1$, $padding = 2$, $pool\_size = 2$

Выход: *output*

1: $output \leftarrow Conv\_layer(input\_tensor, weights_{conv_1}, bias_{conv_1}, stride, padding, padding)$
2: $output \leftarrow Max\_pool(output, pool\_size)$
3: $output \leftarrow ReLU(output)$
4: $output \leftarrow Conv\_layer(output, weights_{conv_2}, bias_{conv_2}, stride, padding)$
5: $output \leftarrow Max\_pool(output, pool\_size)$
6: $output \leftarrow ReLU(output)$
7: линеаризация *output* до размера {64,3136}
8: $output \leftarrow Layer(output, weights_{layer_1}, bias_{layer_1}, 3136, 3136, 3136)$
9: $output \leftarrow ReLU(output)$
10: $output \leftarrow Layer(output, weights_{layer_2}, bias_{layer_2}, 3136, 3136, 3136)$
11: $output \leftarrow ReLU(output)$
12: $output \leftarrow Layer(output, weights_{layer_3}, bias_{layer_3}, 3136, 3136, 3136)$
13: $output \leftarrow ReLU(output)$
14: $output \leftarrow Layer(output, weights_{layer_4}, bias_{layer_4}, 3136, 10, 10)$
15: **Возвратить** *output*

($batch\_size$) равным 64. Размеры сверточного слоя имеют следующий вид $\{64, 1, 8, 8\}$ на входе. Такая расстановка размеров наиболее оптимальна, т.к. при программной разработке СНС в первую очередь обрабатывается размер партии, далее количество каналов и наконец размер изображения. Размеры слоя после применения фильтров зависят, как от размеров сверточного слоя, так и от ядра свертки, в данном случае ядро сверти $k = 5$, количество фильтров – 32 следовательно, учитывая паддинг и пулинг по максимальному элементу равные двум, получим размер выхода равный $\{64, 32, 4, 4\}$. Второй сверточный слой уже имеет размер выхода $\{64, 64, 2, 2\}$. После этого необходимо линеаризовать размерности тензоров до двумерных, чтобы выполнить прямой проход по скрытым слоям. Таким образом, получаем размер $\{64, 256\}$. Следующие три слоя

рассчитываются по формуле (1.1) и их размеры {256, 3136} и {3136, 3136}, размер выходного слоя {3136, 10}. Общий вид СНС-учителя для изображений размером 8×8 представлен на Рисунке 3.2.

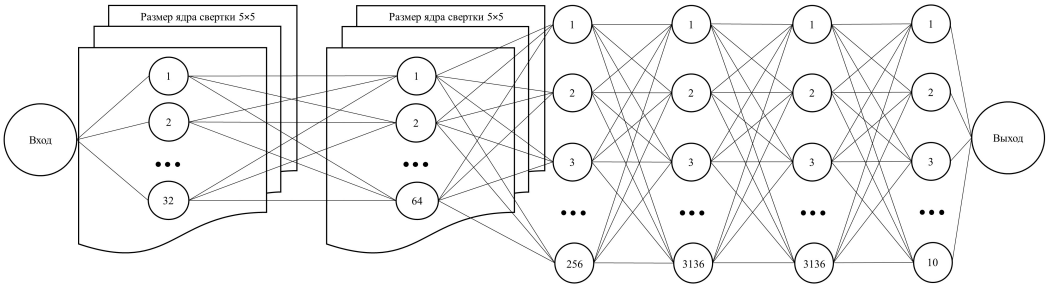


Рисунок 3.2 — Модель СНС-учителя для изображений размером 8×8

Тогда реализацию этой сети можно представить следующим образом (алг. 3.5).

На основе данной математической модели СНС-учителя было обучено 6 моделей СНС. Таким образом, получен ансамбль СНС-учителей для проведения дистилляции. Дистилляция проводится в целях получения СНС малого размера, обеспечивающих точность, сравнимую с точностью сети учителя (в данном случае сетей учителей).

3.1.2 Математические модели сверточных нейронных сетей учеников

Теперь рассмотрим математическую модель СНС-ученика, обученную на наборе изображений в градациях серого (одноканальных) размером 28×28, с объемом партии ($batch\_size$) равным 64. Размеры сверточного слоя имеют следующий вид {64, 1, 28, 28} на входе. Такая расстановка размеров наиболее оптимальна, т.к. при программной разработке СНС в первую очередь обрабатывается размер партии, далее количество каналов и наконец размер изображения. Размеры фильтра зависят, как от размеров сверточного слоя, так и от ядра свертки, в данном случае ядро сверти $k = 7$, количество фильтров – 4 следовательно, учитывая паддинг равный нулю и шаг равный трем, получим размер выхода равный {64, 4, 8, 8}. После этого необходимо линеаризовать размерности тензоров до двумерных, чтобы выполнить прямой проход по скрытым

Алгоритм 3.5. СНС-учитель для изображений 8×8

Вход: $input\_tensor(64, 1, 8, 8)$, $k = 5$, $weights_{conv_1}(1, 32, k, k)$, $bias_{conv_1}(32)$, $weights_{conv_2}(32, 64, k, k)$, $bias_{conv_2}(64)$, $weights_{layer_1}(256, 3136)$, $bias_{layer_1}(3136)$, $weights_{layer_2}(3136, 3136)$, $bias_{layer_2}(3136)$, $weights_{layer_3}(3136, 3136)$, $bias_{layer_3}(3136)$, $weights_{layer_4}(3136, 10)$, $bias_{layer_4}(10)$, $stride = 1$, $padding = 2$, $pool\_size = 2$

Выход: *output*

1: $output \leftarrow Conv\_layer(input\_tensor, weights_{conv_1}, bias_{conv_1}, stride, padding)$
2: $output \leftarrow Max\_pool(output, pool\_size)$
3: $output \leftarrow ReLU(output)$
4: $output \leftarrow Conv\_layer(output, weights_{conv_2}, bias_{conv_2}, stride, padding)$
5: $output \leftarrow Max\_pool(output, pool\_size)$
6: $output \leftarrow ReLU(output)$
7: линеаризация *output* до размера {64,256}
8: $output \leftarrow Layer(output, weights_{layer_1}, bias_{layer_1}, 256, 3136, 3136)$
9: $output \leftarrow ReLU(output)$
10: $output \leftarrow Layer(output, weights_{layer_2}, bias_{layer_2}, 3136, 3136, 3136)$
11: $output \leftarrow ReLU(output)$
12: $output \leftarrow Layer(output, weights_{layer_3}, bias_{layer_3}, 3136, 3136, 3136)$
13: $output \leftarrow ReLU(output)$
14: $output \leftarrow Layer(output, weights_{layer_4}, bias_{layer_4}, 3136, 10, 10)$
15: **Возвратить** *output*

слоям. Таким образом, получаем размер {64, 256}. Первый слой прямого распространения рассчитывается по формуле (1.1) и его размер {256, 64}, размер выходного слоя {64, 10}. Общий вид СНС-ученика для изображений размером 28×28 представлен на Рисунке 3.3.

Данная математическая модель была выбрана исходя из результатов исследований, представленных в Параграфе 2.3. Теперь рассмотрим математическую модель СНС-ученика для изображений размера 8×8, разработанную по подобию модели для изображений 28×28, но с измененным сверточным слоем.

Теперь рассмотрим математическую модель СНС-ученика, обученную на наборе изображений в градациях серого (одноканальных) размером 8×8, с объемом партии ($batch\_size$) равным 64. Размеры сверточного слоя имеют

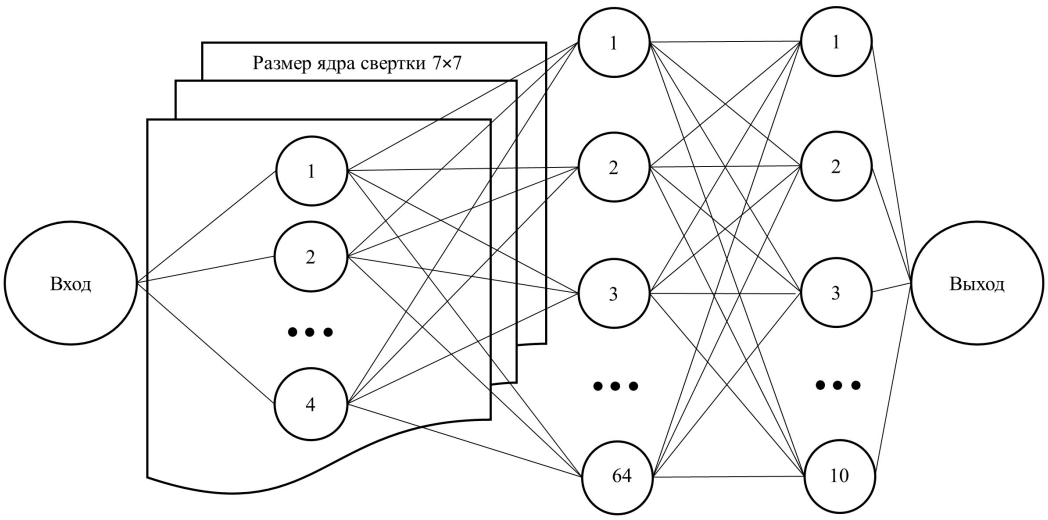


Рисунок 3.3 — Модель СНС-ученика для изображений размером 28×28

следующий вид $\{64,\ 1,\ 8,\ 8\}$ на входе. Такая расстановка размеров наиболее оптимальна, т.к. при программной разработке СНС в первую очередь обрабатывается размер партии, далее количество каналов и наконец размер изображения. Размеры фильтра зависят, как от размеров сверточного слоя, так и от ядра свертки, в данном случае ядро сверти $k = 3$, количество фильтров – 16 следовательно, учитывая паддинг равный нулю и шаг равный одному, получим размер выхода равный $\{64,\ 16,\ 6,\ 6\}$. После этого необходимо линеаризовать размерности тензоров до двумерных, чтобы выполнить прямой проход по скрытым слоям. Таким образом, получаем размер $\{64,\ 576\}$. Первый слой прямого распространения рассчитывается по формуле (1.1) и его размер $\{576,\ 64\}$, размер выходного слоя $\{64,\ 10\}$. Общий вид СНС-ученика для изображений размером 8×8 представлен на Рисунке 3.4.

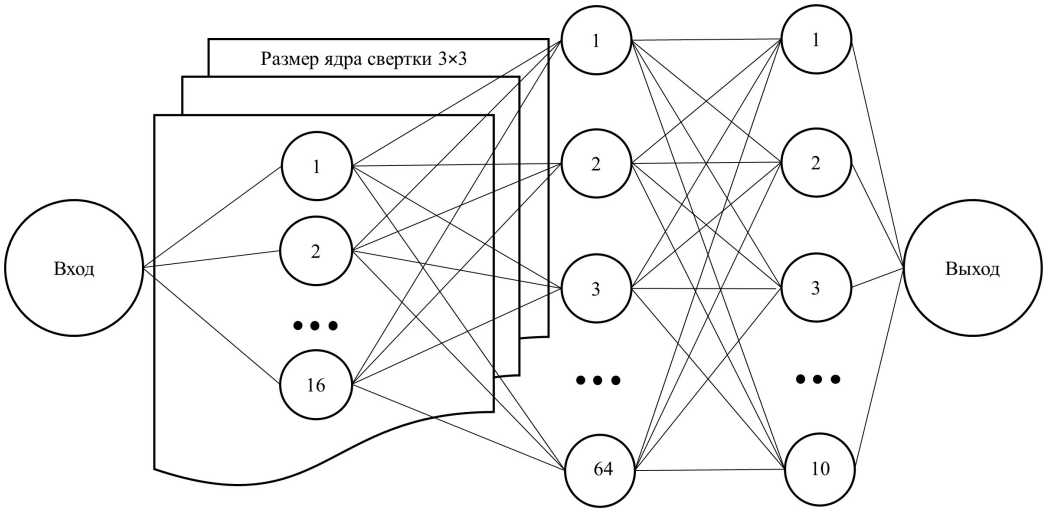


Рисунок 3.4 — Модель СНС-ученика для изображений размером 8×8

В данном параграфе были рассмотрены математические модели СНС, используемых в исследовании, а так же соответствующие алгоритмы. Алгоритмы СНС-учеников будут различаться в зависимости от применяемой функции активации, поэтому они рассмотрены отдельно далее.

3.2 Алгоритмы сверточных нейронных сетей с приближенными функциями активации

3.2.1 Алгоритмы сверточных нейронных сетей с приближенной сигмоидной функцией активации

В данном параграфе рассмотрены СНС, в которых применяются приближенные функции активации. Необходимость в приближении функций активации обусловлена тем, что CKKS не поддерживает операции сравнения и деления; TFHE поддерживает операцию сравнения, операцию деления так же нет. Для исследования зависимости характеристик СНС от функций активации предлагается реализовать и сравнить модели СНС на базе трех приближенных функций активации, а именно ReLU, ELU и сигмоидной. Для приближенной сигмоидной функции активации (ПС) в работе [116] получен полином вида

$$\hat{\sigma}(x) = -0.004 \cdot x^3 + 0.197 \cdot x + 0.5. \tag{3.1}$$

Алгоритм реализации такой функции активации достаточно прост (алг. 3.6).

Алгоритм 3.6. ПС (функция *Sigmoid_Approximate_Activation*)

Вход: x

Выход: sig

1: $sig = -0.004 \cdot x^3 + 0.197 \cdot x + 0.5$

2: **Возвратить** sig

Тогда реализация СНС с ПС (СНСПС) для изображений 28×28 будет иметь следующий вид (алг. 3.7).

Реализация СНСПС для изображений 8×8 представлена Алгоритмом 3.8.

Алгоритм 3.7. СНСПС для изображений размером 28×28

Вход: $input\_tensor(64,\ 1,\ 28,\ 28)$, $k = 7$, $weights_{conv_1}(1,\ 4,\ k,\ k)$, $bias_{conv_1}(4)$, $weights_{layer_1}(256,\ 64)$, $bias_{layer_1}(64)$, $weights_{layer_2}(64,\ 10)$, $bias_{layer_2}(10)$, $stride = 3$, $padding = 0$

Выход: *output*

1: $output \leftarrow Conv\_layer(input\_tensor, weights_{conv_1}, bias_{conv_1}, stride, padding)$
2: $output \leftarrow Sigmoid\_Approximate\_Activation(output)$
3: линеаризация *output* до размера $\{64, 256\}$
4: $output \leftarrow Layer(output, weights_{layer_1}, bias_{layer_1}, 256, 64, 64)$
5: $output \leftarrow Sigmoid\_Approximate\_Activation(output)$
6: $output \leftarrow Layer(output, weights_{layer_2}, bias_{layer_2}, 64, 10, 10)$
7: **Возвратить** *output*

Алгоритм 3.8. СНСПС для изображений размером 8×8

Вход: $input\_tensor(64,\ 1,\ 8,\ 8)$, $k = 3$, $weights_{conv_1}(1,\ 16,\ k,\ k)$, $bias_{conv_1}(4)$, $weights_{layer_1}(576,\ 64)$, $bias_{layer_1}(64)$, $weights_{layer_2}(64,\ 10)$, $bias_{layer_2}(10)$, $stride = 1$, $padding = 0$

Выход: *output*

1: $output \leftarrow Conv\_layer(input\_tensor, weights_{conv_1}, bias_{conv_1}, stride, padding)$
2: $output \leftarrow Sigmoid\_Approximate\_Activation(output)$
3: линеаризация *output* до размера $\{64, 576\}$
4: $output \leftarrow Layer(output, weights_{layer_1}, bias_{layer_1}, 576, 64, 64)$
5: $output \leftarrow Sigmoid\_Approximate\_Activation(output)$
6: $output \leftarrow Layer(output, weights_{layer_2}, bias_{layer_2}, 64, 10, 10)$
7: **Возвратить** *output*

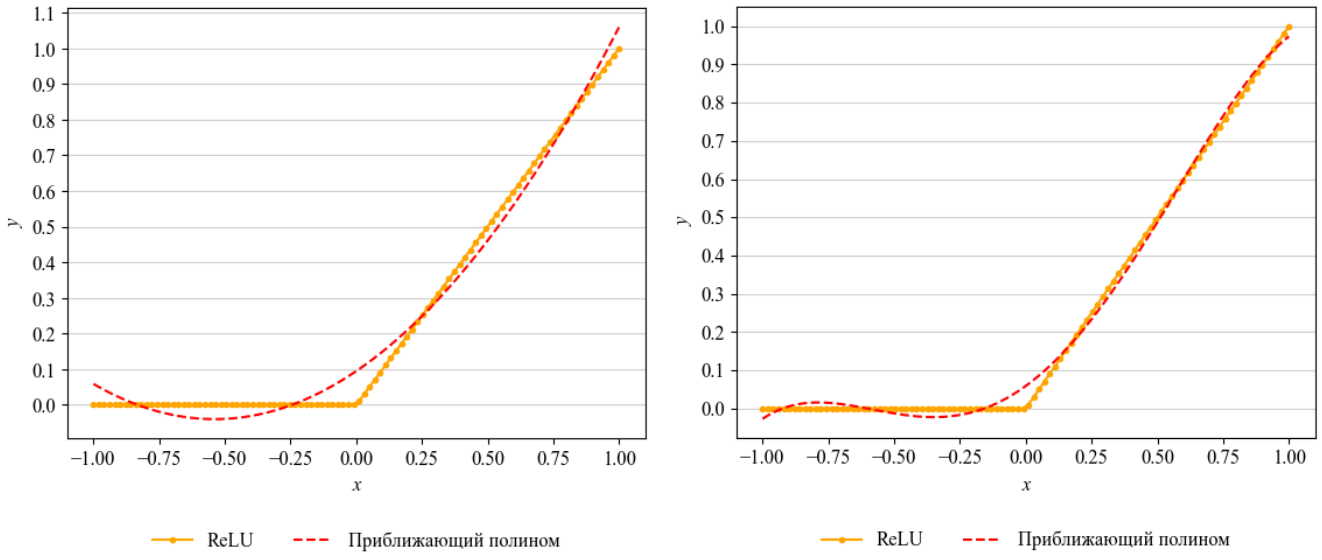
Таким образом, получены реализации СНСПС для двух наборов изображений разного размера; они будут применяться как для обучения с дистилляцией, так и без нее. Далее рассмотрим СНС с приближенной функцией активации ReLU.

3.2.2 Алгоритмы сверточных нейронных сетей с приближенной функцией активации ReLU

В рамках проводимых исследований для ПР было простроено несколько приближающих полиномов, а именно полиномы 2, 3, 4 и 5 степеней. Полиномы имеют следующий вид:

$$\begin{aligned}
\hat{y}_2 &= 0.0464108 \cdot x^2 + 0.5 \cdot x + 0.946969\,, \\
\hat{y}_3 &= 5.931968 \cdot 10^{-18} \cdot x^3 + 0.0464108 \cdot x^2 + 0.5 \cdot x + 0.946969\,, \\
\hat{y}_4 &= -3.983725 \cdot 10^{-4} \cdot x^4 + 1.824722 \cdot 10^{-18} \cdot x^3 + 0.0812353 \cdot x^2 + \\
&\quad + 0.5 \cdot x + 0.591856\,, \\
\hat{y}_5 &= 1.112156 \cdot 10^{-19} \cdot x^5 - 3.983725 \cdot 10^{-4} \cdot x^4 - 1.676463 \cdot 10^{-17} \cdot x^3 + \\
&\quad + 0.0812353 \cdot x^2 + 0.5 \cdot x + 0.591856\,.
\end{aligned} \tag{3.2}$$

Можно заметить, что полиномы 2 и 3 степеней, а так же 4 и 5 степеней практически не отличаются, поэтому достаточно рассмотреть полиномы 2 и 4 степеней.



а) б)

Рисунок 3.5 — Графики ПР: а) Вторая степень; б) Четвертая степень

Полиномы степени больше 5 рассматривать в данном исследовании достаточно проблематично ввиду того, что потребуется еще большее количество умножений, что затруднит применение схем ПГШ. Если обратиться к Рисунку 3.5, можно заметить, что хуже всего приближение функции происходит в

окрестности точки $(0, 0)$. Кроме того, если проанализировать коэффициент детерминации, который вычисляется по формуле

$$R^2 = \frac{\sum_{i=1}^{N} (\hat{y}_i - \bar{y})^2}{\sum_{i=1}^{N} (y_i - \bar{y})^2}, \tag{3.3}$$

где $\bar{y} = \frac{1}{N} \sum_{j=1}^{N} y_j$, N – длина вектора y, можно получить следующие значения R^2 для построенных полиномов 2 и 4 степеней

$$\begin{aligned} R^2_{\hat{y}_2} &= 0.987521, \\ R^2_{\hat{y}_4} &= 0.996888. \end{aligned} \tag{3.4}$$

Полиномы характеризуются достаточно высокой точностью приближения, поэтому в дальнейших исследованиях будут использованы они оба. Алгоритм 3.9 реализует данную функцию активации.

Алгоритм 3.9. ПР (функция *ReLU_Approximate_Activation*)

Вход: x, $flag$

Выход: *output*

1: **Если** $flag = 2$ **тогда**

2: $\quad output = 0.0464108 \cdot x^2 + 0.5 \cdot x + 0.946969$

3: **Конец условия**

4: **Если** $flag = 4$ **тогда**

5: $\quad output = -3.983725 \cdot 10^{-4} \cdot x^4 + 0.0812353 \cdot x^2 + 0.5 \cdot x + 0.591856$

6: **Конец условия**

7: **Возвратить** *output*

Тогда реализация СНС с ПР (СНСПР) для изображений размера 28×28 имеет следующий вид (алг. 3.10).

Алгоритм 3.11 реализует СПСПР для изображений размера 8×8.

Таким образом, получены реализации СНСПР для двух наборов изображений разного размера; они будут применяться как для обучения с дистилляцией, так и без нее.

Алгоритм 3.10. СНСПР для изображений размером 28×28

Вход: $input\_tensor(64,\ 1,\ 28,\ 28)$, $k = 7$, $weights_{conv_1}(1,\ 4,\ k,\ k)$, $bias_{conv_1}(4)$, $weights_{layer_1}(256,\ 64)$, $bias_{layer_1}(64)$, $weights_{layer_2}(64,\ 10)$, $bias_{layer_2}(10)$, $stride = 3$, $padding = 0$, $flag$

Выход: *output*

1: $output \leftarrow Conv\_layer(input\_tensor, weights_{conv_1}, bias_{conv_1}, stride, padding)$
2: $output \leftarrow ReLU\_Approximate\_Activation(output,\ flag)$
3: линеаризация *output* до размера {64, 256}
4: $output \leftarrow Layer(output, weights_{layer_1}, bias_{layer_1},\ 256,\ 64,\ 64)$
5: $output \leftarrow ReLU\_Approximate\_Activation(output,\ flag)$
6: $output \leftarrow Layer(output, weights_{layer_2}, bias_{layer_2},\ 64,\ 10,\ 10)$
7: **Возвратить** *output*

Алгоритм 3.11. СНСПР для изображений размером 8×8

Вход: $input\_tensor(64,\ 1,\ 8,\ 8)$, $k = 3$, $weights_{conv_1}(16,\ 1,\ k,\ k)$, $bias_{conv_1}(16)$, $weights_{layer_1}(576,\ 64)$, $bias_{layer_1}(64)$, $weights_{layer_2}(64,\ 10)$, $bias_{layer_2}(10)$, $stride = 1$, $padding = 0$, $flag$

Выход: *output*

1: $output \leftarrow Conv\_layer(input\_tensor, weights_{conv_1}, bias_{conv_1}, stride, padding)$
2: $output \leftarrow ReLU\_Approximate\_Activation(output,\ flag)$
3: линеаризация *output* до размера {64, 576}
4: $output \leftarrow Layer(output,\ weights_{layer_1},\ bias_{layer_1},\ 576,\ 64,\ 64)$
5: $output \leftarrow ReLU\_Approximate\_Activation(output,\ flag)$
6: $output \leftarrow Layer(output,\ weights_{layer_2}, bias_{layer_2},\ 64,\ 10,\ 10)$
7: **Возвратить** *output*

3.2.3 Алгоритмы сверточных нейронных сетей с приближенной функцией активации ELU

ReLU является одной из наиболее часто используемых функций активации, в особенности при разработке СНС. В Параграфе 2.3 были рассмотрены несколько вариаций функции активации ReLU, наиболее же интересные результаты были получены при построении СНС с функцией активации ELU, поэтому рассмотрим подробнее данную функцию активации.

Как и в случае с ReLU, для ПЕ было простроено несколько полиномов, а именно полиномы 2, 3, 4 и 5 степеней. Полиномы имеют следующий вид:

$$\begin{aligned}
\hat{y}_2 &= 0.0447007 \cdot x^2 + 0.5728 \cdot x + 0.554538\,, \\
\hat{y}_3 &= -0.000949 \cdot x^3 + 0.0447007 \cdot x^2 + 0.630903 \cdot x + 0.554538\,, \\
\hat{y}_4 &= -3.40065 \cdot 10^{-4} \cdot x^4 - 9.493416 \cdot 10^{-4} \cdot x^3 + 0.074428 \cdot x^2 + \\
&\quad + 0.630903 \cdot x + 0.2514\,, \\
\hat{y}_5 &= 1.99471 \cdot 10^{-5} \cdot x^5 - 3.40065 \cdot 10^{-4} \cdot x^4 - 3.209105 \cdot 10^{-3} \cdot x^3 + \\
&\quad + 0.074428 \cdot x^2 + 0.680269 \cdot x + 0.2514\,.
\end{aligned} \tag{3.5}$$

Проанализируем графики ПЕ (рис. 3.6).

В отличие от ReLU, все 4 полученных полинома различаются. Кроме того, в данном случае отсутствуют нулевые коэффициенты для членов полинома. Получим значения R^2 по формуле (3.3).

$$\begin{aligned}
R^2_{\hat{y}_2} &= 0.990918, \\
R^2_{\hat{y}_3} &= 0.992582, \\
R^2_{\hat{y}_4} &= 0.996888, \\
R^2_{\hat{y}_5} &= 0.998597.
\end{aligned} \tag{3.6}$$

Поскольку удалось добиться хорошего приближения, далее в исследовании будут использоваться все четыре полученных полинома. Алгоритм 3.12 реализует данную функцию активации.

Теперь можно использовать разработанный алгоритм в реализации СНС с ПЕ (СНСПЕ). Для изображений размера 28×28, реализация представлена Алгоритмом 3.13.

Алгоритм 3.14 реализует СПСПЕ для изображений размера 8×8.

Таким образом, получены реализации СНСПЕ для двух наборов изображений разного размера; они будут применяться далее, как для обучения с дистилляцией, так и без нее.

Алгоритм 3.12. ПЕ (функция *ELU_Approximate_Activation*)

Вход: x, $flag$

Выход: *output*

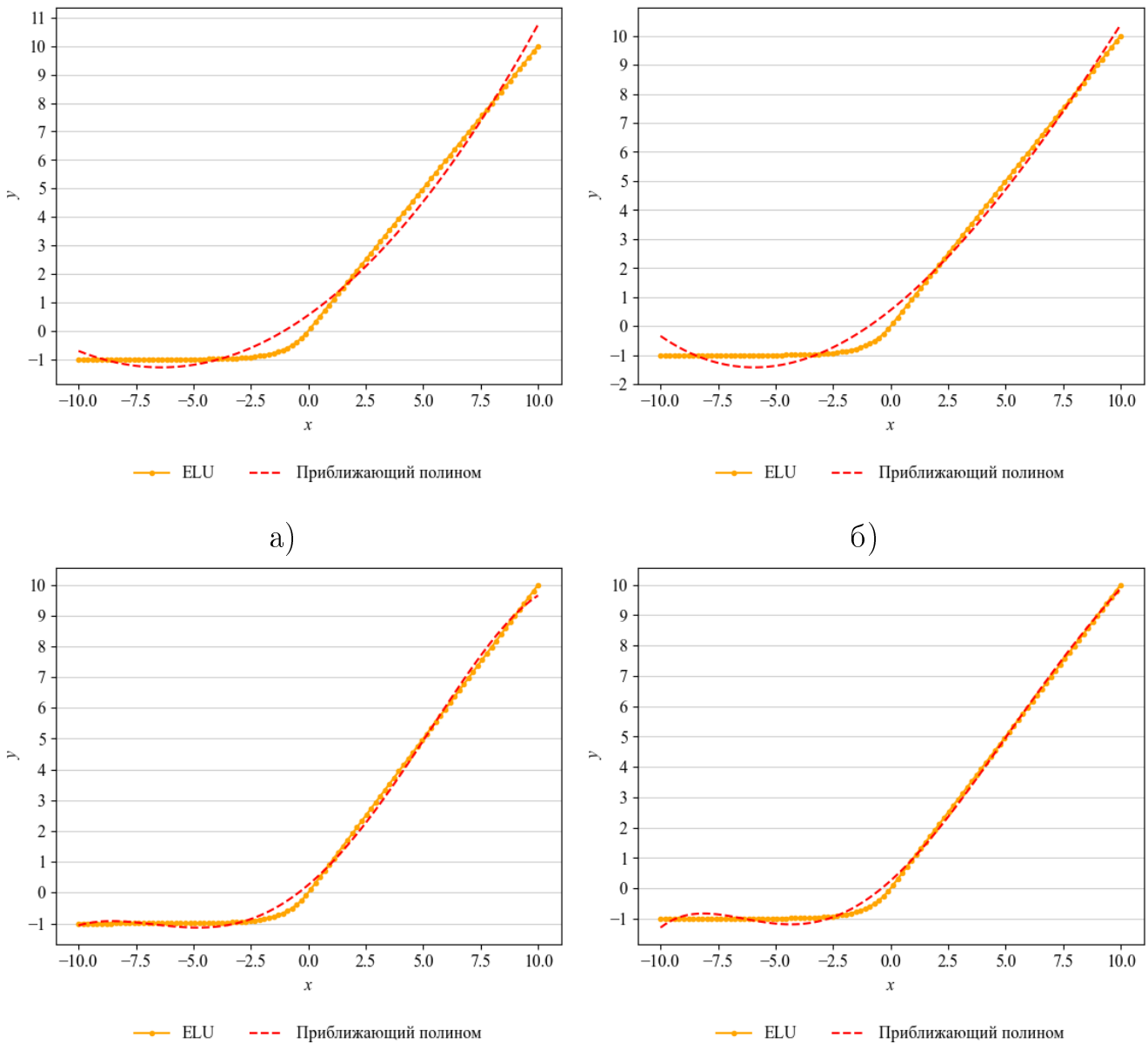
Если $flag = 2$ **тогда**
    $output = 0.0447007 \cdot x^2 + 0.5728 \cdot x + 0.554538$
Конец условия
Если $flag = 3$ **тогда**
    $output = -0.000949 \cdot x^3 + 0.0447007 \cdot x^2 + 0.630903 \cdot x + 0.554538$
Конец условия
Если $flag = 4$ **тогда**
    $output = -3.40065 \cdot 10^{-4} \cdot x^4 - 9.493416 \cdot 10^{-4} \cdot x^3 + 0.074428 \cdot x^2 + 0.630903 \cdot x + 0.2514$
Конец условия
Если $flag = 5$ **тогда**
    $output = 1.99471 \cdot 10^{-5} \cdot x^5 - 3.40065 \cdot 10^{-4} \cdot x^4 - 3.209105 \cdot 10^{-3} \cdot x^3 + 0.074428 \cdot x^2 + 0.680269 \cdot x + 0.2514$
Конец условия
Возвратить *output*

Алгоритм 3.13. СНСПЕ для изображений размером 28×28

Вход: $input\_tensor(64,\ 1,\ 28,\ 28)$, $k = 7$, $weights_{conv_1}(1,\ 4,\ k,\ k)$, $bias_{conv_1}(4)$, $weights_{layer_1}(256,\ 64)$, $bias_{layer_1}(64)$, $weights_{layer_2}(64,\ 10)$, $bias_{layer_2}(10)$, $stride = 3$, $padding = 0$, $flag$

Выход: *output*

$output \leftarrow Conv\_layer(input\_tensor, weights_{conv_1}, bias_{conv_1}, stride, padding)$
$output \leftarrow ELU\_Approximate\_Activation(output,\ flag)$
линеаризация *output* до размера $\{64,\ 256\}$
$output \leftarrow Layer(output, weights_{layer_1}, bias_{layer_1},\ 256,\ 64,\ 64)$
$output \leftarrow ELU\_Approximate\_Activation(output,\ flag)$
$output \leftarrow Layer(output, weights_{layer_2}, bias_{layer_2},\ 64,\ 10,\ 10)$
Возвратить *output*



а) б)

в) г)

Рисунок 3.6 — Графики ПЕ: а) Вторая степень; б) Третья степень; в) Четвертая степень; г) Пятая степень

3.2.4 Алгоритмы сверточных нейронных сетей с квадратичной функцией активации

Разработчики TenSEAL в документации к библиотеке использовали КФА вида x^2. Поэтому для полноты исследования так же рассмотрим реализации СНС на базе данной функции активации. Реализация СНС с КФА (СНСКФА) для изображений размера 28×28 представлена Алгоритмом 3.15.

Алгоритм 3.16 реализует СНСКФА для изображений размера 8×8.

Алгоритм 3.14. СПСПЕ для изображений размером 8×8

Вход: $input\_tensor(64,\ 1,\ 8,\ 8)$, $k = 3$, $weights_{conv_1}(1,\ 16,\ k,\ k)$, $bias_{conv_1}(16)$, $weights_{layer_1}(576,\ 64)$, $bias_{layer_1}(64)$, $weights_{layer_2}(64,\ 10)$, $bias_{layer_2}(10)$, $stride = 1$, $padding = 0$, $flag$

Выход: $output$

1: $output \leftarrow Conv\_layer(input\_tensor, weights_{conv_1}, bias_{conv_1}, stride, padding)$
2: $output \leftarrow ELU\_Approximate\_Activation(output,\ flag)$
3: линеаризация $output$ до размера $\{64,\ 576\}$
4: $output \leftarrow Layer(output,\ weights_{layer_1},\ bias_{layer_1},\ 576,\ 64,\ 64)$
5: $output \leftarrow ELU\_Approximate\_Activation(output,\ flag)$
6: $output \leftarrow Layer(output,\ weights_{layer_2}, bias_{layer_2},\ 64,\ 10,\ 10)$
7: **Возвратить** $output$

Алгоритм 3.15. СНСКФА для изображений размером 28×28

Вход: $input\_tensor(64,\ 1,\ 28,\ 28)$, $k = 7$, $weights_{conv_1}(1,\ 4,\ k,\ k)$, $bias_{conv_1}(4)$, $weights_{layer_1}(256,\ 64)$, $bias_{layer_1}(64)$, $weights_{layer_2}(64,\ 10)$, $bias_{layer_2}(10)$, $stride = 3$, $padding = 0$

Выход: $output$

1: $output \leftarrow Conv\_layer(input\_tensor, weights_{conv_1}, bias_{conv_1}, stride, padding)$
2: $output \leftarrow output \cdot output$
3: линеаризация $output$ до размера $\{64,\ 256\}$
4: $output \leftarrow Layer(output, weights_{layer_1}, bias_{layer_1},\ 256,\ 64,\ 64)$
5: $output \leftarrow output \cdot output$
6: $output \leftarrow Layer(output, weights_{layer_2}, bias_{layer_2},\ 64,\ 10,\ 10)$
7: **Возвратить** $output$

Таким образом, получены реализации СНСКФА для обоих наборов изображений разного размера. Как и в случае с другими моделями, они будут применяться как для обучения с дистилляцией, так и без нее.

Алгоритм 3.16. СНСКФА для изображений размером 8×8

Вход: $input\_tensor(64,\ 1,\ 8,\ 8)$, $k = 3$, $weights_{conv_1}(1,\ 16,\ k,\ k)$, $bias_{conv_1}(16)$, $weights_{layer_1}(576,\ 64)$, $bias_{layer_1}(64)$, $weights_{layer_2}(64,\ 10)$, $bias_{layer_2}(10)$, $stride = 1$, $padding = 0$

Выход: *output*

1: $output \leftarrow Conv\_layer(input\_tensor, weights_{conv_1}, bias_{conv_1}, stride, padding)$
2: $output \leftarrow output \cdot output$
3: линеаризация *output* до размера $\{64,\ 576\}$
4: $output \leftarrow Layer(output,\ weights_{layer_1},\ bias_{layer_1},\ 576,\ 64,\ 64)$
5: $output \leftarrow output \cdot output$
6: $output \leftarrow Layer(output,\ weights_{layer_2}, bias_{layer_2},\ 64,\ 10,\ 10)$
7: **Возвратить** *output*

3.2.5 Алгоритмы сверточных нейронных сетей с функцией активации с обучаемыми коэффициентами

Вернемся к функции активации ФАОК из Параграфа 2.3.2. Рассмотрим выражение (2.12). Учитывая, что коэффициенты α обучаются так же, как и веса нейронов, можно построить схожий алгоритм. Реализация функции активации ФАОК представлена Алгоритмом 3.17).

Алгоритм 3.17. ФАОК
(функция $Activation\_Functuon\_Learning\_Coefficient$)

Вход: x, α

Выход: *output*

1: $output = 0$
2: **Цикл от** $i = 0$ **до** $length(x) - 1$ **выполнять**
3: $\quad output \leftarrow output + \alpha_i \cdot output^i$
4: **Конец цикла**
5: **Возвратить** *output*

Теперь можно использовать разработанный алгоритм в реализациях СНС с функцией активации ФАОК (СНСФАОК). Для изображений размера 28×28, реализация представлена Алгоритмом 3.18.

Алгоритм 3.19 реализует СНСФАОК для изображений размера 8×8.

Алгоритм 3.18. СНСФАОК для изображений размером 28×28

Вход: $input\_tensor(64,\ 28,\ 28)$, $k = 7$, $weights_{conv_1}(4,\ 1,\ k,\ k)$, $bias_{conv_1}(4)$, $weights_{layer_1}(256,\ 64)$, $bias_{layer_1}(64)$, $weights_{layer_2}(64,\ 10)$, $bias_{layer_2}(10)$, $stride = 3$, $padding = 0$, α

Выход: *output*

1: $output \leftarrow Conv\_layer(input\_tensor, weights_{conv_1}, bias_{conv_1}, stride, padding)$
2: $output \leftarrow Activation\_Functuon\_Learning\_Coefficient(output, \alpha)$
3: линеаризация *output* до размера {64, 256}
4: $output \leftarrow Layer(output, weights_{layer_1}, bias_{layer_1}, 256, 64, 64)$
5: $output \leftarrow Activation\_Functuon\_Learning\_Coefficient(output, \alpha)$
6: $output \leftarrow Layer(output, weights_{layer_2}, bias_{layer_2}, 64, 10, 10)$
7: **Возвратить** *output*

Алгоритм 3.19. СНСФАОК для изображений размером 8×8

Вход: $input\_tensor(64,\ 8,\ 8)$, $k = 3$, $weights_{conv_1}(16,\ 1,\ k,\ k)$, $bias_{conv_1}(16)$, $weights_{layer_1}(576,\ 64)$, $bias_{layer_1}(64)$, $weights_{layer_2}(64,\ 10)$, $bias_{layer_2}(10)$, $stride = 1$, $padding = 0$, α_{conv}, α_{layer}

Выход: *output*

1: $output \leftarrow Conv\_layer(input\_tensor, weights_{conv_1}, bias_{conv_1}, stride, padding)$
2: $output \leftarrow Activation\_Functuon\_Learning\_Coefficient(output,\ \alpha_{conv})$
3: линеаризация *output* до размера {64, 576}
4: $output \leftarrow Layer(output,\ weights_{layer_1},\ bias_{layer_1},\ 576, 64, 64)$
5: $output \leftarrow Activation\_Functuon\_Learning\_Coefficient(output,\ \alpha_{layer})$
6: $output \leftarrow Layer(output,\ weights_{layer_2}, bias_{layer_2},\ 64, 10, 10)$
7: **Возвратить** *output*

Реализованные модели далее используются в качестве основы для построения СНС с применением ПГШ. Учитывая тот факт, что в данных алгоритмах применяются исключительно операции сложения и умножения, построение моделей СНССК не составит труда. Реализация сверточного, слоя сохраняющего конфиденциальность, была представлена в Параграфе 2.4, а метод умножения, сохраняющего конфиденциальность, описан в Параграфе 2.2. Полученные результаты будут использованы для разработки математических моделей СНССК как на базе CKKS, так и на базе TFHE, но с учетом того, что операции вращения в TFHE не характерны для CKKS.

3.3 Математические модели нейронных сетей сохраняющих конфиденциальность

Реализации сверточного и скрытого слоев были представлены в Параграфе 2.4.5 (алг. 2.8, алг. 2.9). Принимая во внимание разработанный метод умножения (теор. 1), можно говорить о том, что данные алгоритмы подходят, как для схемы CKKS, так и для схемы TFHE. В случае схемы CKKS выигрыш заключается в уменьшении потребления памяти и количества гомоморфных умножений; в случае схемы TFHE, выигрыш заключается в уменьшении потребления памяти и сокращении количества операций слепого вращения. Аналогично концепции Параграфа 3.2, в данном параграфе получены математические модели CHCCK с приближенными функциями активации ReLU, ELU, приближенной сигмоидной функцией, с квадратичной функцией активации и функцией активации ФАОК. Так же для TFHE будут рассмотрены функции активации ReLU и ELU заданные в явном виде, так как схема TFHE поддерживает сравнение. Для удобства разделим алгоритмы на две группы:

1. Алгоритмы для изображений размером 28×28.
2. Алгоритмы для изображений размером 8×8.

Начнем с рассмотрения приближенных функции активации, обрабатывающих данные в зашифрованном виде. Рассмотрим зашифрованную приближенную сигмоидную функцию активации (алг. 3.20), под *context* подразумеваются параметры шифрования.

Теперь рассмотрим Алгоритм 3.21, реализующий приближенную зашифрованную функцию активации ReLU.

В целом зашифрованная ПР, как и сигмоидная функция, отличается от открытого варианта только способом умножения и сложения значений, а так же необходимостью применения параметров шифрования. Аналогичным образом построим Алгоритм 3.22 для зашифрованной ПЕ.

Для квадратичной функции активации описывать отдельный алгоритм не имеет смысла, так как она представляет собой умножение зашифрованного вектора x_{enc} самого на себя, т.е. функцию $\mathrm{Mult}(x, x, context)$, которая и будет прописана в соответствующих реализациях моделей CHCCK.

Теперь рассмотрим алгоритм, реализующий зашифрованную функцию активации ФАОК. Алгоритм будет выглядеть следующим образом (алг. 3.23).

Алгоритм 3.20. Зашифрованная ПС
(функция $Sigmoid\_Approximate\_Activation_{enc}$)

Вход: x, $context$

Выход: $output$

1: $\text{coefficient} = [0.5, 0.197, 0, -0.004]$
2: $output \leftarrow Encryption\_Context(0, context)$
3: $temp \leftarrow Encryption\_Context(1, context)$
4: **Цикл от** $i = 0$ **до** $length(\text{coefficient}) - 1$ **выполнять**
5: $\quad output \leftarrow Add(output, Mult_{Plain}(temp, \text{coefficient}_i, context), context)$
6: $\quad temp \leftarrow Mult(temp, x, context)$
7: **Конец цикла**
8: **Возвратить** $output$

Алгоритм 3.21. Зашифрованная ПР
(функция $ReLU\_Approximate\_Activation_{enc}$)

Вход: x, $flag$, $context$

Выход: $output$

1: $\text{coefficient}_2 = [0.9469697, 0.5, 0.04641089]$, $\text{coefficient}_4 = [0.591856, 0.5, 0.0812353, 0, -3.983725 \cdot 10^{-4}]$
2: $output \leftarrow Encryption\_Context(0, context)$
3: $temp \leftarrow Encryption\_Context(1, context)$
4: **Цикл от** $i = 0$ **до** $length(\text{coefficient}_{flag}) - 1$ **выполнять**
5: $\quad output \leftarrow Add(output, Mult_{Plain}(temp, \text{coefficient}_{flag_i}, context), context)$
6: $\quad temp \leftarrow Mult(temp, x, context)$
7: **Конец цикла**
8: **Возвратить** $output$

Рассмотрим СНССК с зашифрованной приближенной сигмоидной функцией активации. Воспользуемся Алгоритмами 2.6-2.9, 3.20 и получим Алгоритм 3.24, реализующий СНСПС сохраняющую конфиденциальность (СНСПССК).

Алгоритм 3.22. Зашифрованная ПЕ
(функция $ELU\_Approximate\_Activation_{enc}$)

Вход: x, $flag$, $context$

Выход: $output$

1: $\text{coefficient}_2 = [0.554538, 0.5728, 0.0447007]$, $\text{coefficient}_3 = [0.554538, 0.630903, 0.0447007, -0.000949]$, $\text{coefficient}_4 = [0.2514, 0.630903, 0.074428, -9.493416 \cdot 10^{-4}, -3.40065 \cdot 10^{-4}]$, $\text{coefficient}_5 = [0.2514, 0.680269, 0.074428, -3.209105 \cdot 10^{-3}, -3.40065 \cdot 10^{-4}, 1.99471 \cdot 10^{-5}]$
2: $output \leftarrow Encryption\_Context(0, context)$
3: $temp \leftarrow Encryption\_Context(1, context)$
4: **Цикл от** $i = 0$ **до** $length(\text{coefficient}_{flag}) - 1$ **выполнять**
5: $\quad output \leftarrow Add(output, Mult_{Plain}(temp, \text{coefficient}_{flag_i}, context), context)$
6: $\quad temp \leftarrow Mult(temp, x, context)$
7: **Конец цикла**
8: **Возвратить** $output$

Алгоритм 3.23. Зашифрованная ФАОК
(Функция $Activation\_Function\_Learning\_\text{Coefficient}_{enc}$)

Вход: x, coefficient, $context$

Выход: $output$

1: $output \leftarrow Encryption\_Context(0, context)$
2: $temp \leftarrow Encryption\_Context(1, context)$
3: **Цикл от** $i = 0$ **до** $length(\text{coefficient}) - 1$ **выполнять**
4: $\quad output \leftarrow Add(output, Mult_{Plain}(temp, \text{coefficient}_i, context), context)$
5: $\quad temp \leftarrow Mult(temp, x, context)$
6: **Конец цикла**
7: **Возвратить** $output$

Алгоритм 3.24. СНСПССК

Вход: $image_{enc}$, k, $weights_{conv}$, $bias_{conv}$, $weights_{layer_1}$, $bias_{layer_1}$, $weights_{layer_2}$, $bias_{layer_2}$, $stride$, $context$

Выход: $output$

1: $output \leftarrow Layer_{conv_{enc}}(image_{enc}, k, stride, weights_{conv}, bias_{conv}, context)$
2: $output \leftarrow Sigmoid\_Approximate\_Activation_{enc}(output, context)$
3: линеаризация $output$ до размера $\{64, RANGE^*\}$
4: $output \leftarrow Encryption\_Forward\_Propagation(output, weights_{layer_1}, bias_{layer_1}, context)$
5: $output \leftarrow Sigmoid\_Approximate\_Activation_{enc}(output, context)$
6: $output \leftarrow Encryption\_Forward\_Propagation(output, weights_{layer_2},$

Аналогичный Алгоритм 3.25 получаем для реализации СНСПР сохраняющей конфиденциальность (СНСПРСК).

Алгоритм 3.25. СНСПРСК

Вход: $image_{enc}$, k, $weights_{conv}$, $bias_{conv}$, $weights_{layer_1}$, $bias_{layer_1}$, $weights_{layer_2}$, $bias_{layer_2}$, $stride$, $flag$, $context$

Выход: $output$

1: $output \leftarrow Layer_{conv_{enc}}(image_{enc}, k, stride, weights_{conv}, bias_{conv}, context)$

2: $output \leftarrow ReLU\_Approximate\_Activation_{enc}(output, flag, context)$

3: линеаризация $output$ до размера $\{64, RANGE^*\}$

4: $output \leftarrow Encryption\_Forward\_Propagation(output, weights_{layer_1}, bias_{layer_1}, context)$

5: $output \leftarrow ReLU\_Approximate\_Activation_{enc}(output, flag, context)$

6: $output \leftarrow Encryption\_Forward\_Propagation(output, weights_{layer_2}, bias_{layer_2}, context)$

7: **Возвратить** $output$

▷ $*$ – Для размера 28×28 – $RANGE = 256$, для размера 8×8 – $RANGE = 576$

Алгоритм 3.26 реализует СНСПЕ сохраняющую конфиденциальность (СНСПЕСК).

Алгоритм 3.26. СНСПЕСК

Вход: $image_{enc}$, k, $weights_{conv}$, $bias_{conv}$, $weights_{layer_1}$, $bias_{layer_1}$, $weights_{layer_2}$, $bias_{layer_2}$, $stride$, $flag$, $context$

Выход: $output$

1: $output \leftarrow Layer_{conv_{enc}}(image_{enc}, k, stride, weights_{conv}, bias_{conv}, context)$

2: $output \leftarrow ELU\_Approximate\_Activation_{enc}(output, flag, context)$

3: линеаризация $output$ до размера $\{64, RANGE^*\}$

4: $output \leftarrow Encryption\_Forward\_Propagation(output, weights_{layer_1}, bias_{layer_1}, context)$

5: $output \leftarrow ELU\_Approximate\_Activation_{enc}(output, flag, context)$

6: $output \leftarrow Encryption\_Forward\_Propagation(output, weights_{layer_2}, bias_{layer_2}, context)$

7: **Возвратить** $output$

▷ $*$ – Для размера 28×28 – $RANGE = 256$, для размера 8×8 – $RANGE = 576$

СНСКФА сохраняющую конфиденциальность (СНСКФАСК) реализует Алгоритм 3.27.

Алгоритм 3.27. СНСКФАСК

Вход: $image_{enc}$, k, $weights_{conv}$, $bias_{conv}$, $weights_{layer_1}$, $bias_{layer_1}$, $weights_{layer_2}$, $bias_{layer_2}$, $stride$, $context$

Выход: $output$

1: $output \leftarrow Layer_{conv_{enc}}(image_{enc}, k, stride, weights_{conv}, bias_{conv}, context)$

2: $output \leftarrow Mult(output, output, context)$

3: линеаризация $output$ до размера $\{64, RANGE^*\}$

4: $output \leftarrow Encryption\_Forward\_Propagation(output, weights_{layer_1}, bias_{layer_1}, context)$

5: $output \leftarrow Mult(output, output, context)$

6: $output \leftarrow Encryption\_Forward\_Propagation(output, weights_{layer_2}, bias_{layer_2}, context)$

7: **Возвратить** $output$

▷ $*$ – Для размера 28×28 – $RANGE = 256$, для размера 8×8 – $RANGE = 576$

Наконец, Алгоритм 3.28 реализует СНСФАОК сохраняющую конфиденциальность (СНСФАОКСК).

Алгоритм 3.28. СНСФАОКСК

Вход: $image_{enc}$, k, $weights_{conv}$, $bias_{conv}$, $weights_{layer_1}$, $bias_{layer_1}$, $weights_{layer_2}$, $bias_{layer_2}$, $stride$, α_{conv}, α_{layer}, $context$

Выход: $output$

1: $output \leftarrow Layer_{conv_{enc}}(image_{enc}, k, stride, weights_{conv}, bias_{conv}, context)$

2: $output \leftarrow Activation\_Functuon\_Learning\_Coefficients_{enc}(output, \alpha_{conv}, context)$

3: линеаризация $output$ до размера $\{64, RANGE^*\}$

4: $output \leftarrow Encryption\_Forward\_Propagation(output, weights_{layer_1}, bias_{layer_1}, context)$

5: $output \leftarrow Activation\_Functuon\_Learning\_Coefficients_{enc}(output, \alpha_{layer}, context)$

6: $output \leftarrow Encryption\_Forward\_Propagation(output, weights_{layer_2}, bias_{layer_2}, context)$

7: **Возвратить** $output$

▷ $*$ – Для размера 28×28 – $RANGE = 256$, для размера 8×8 – $RANGE = 576$

В итоге были получены алгоритмы, реализующие СНССК с 5 различными функциями активации. Реализации для наборов изображений размерами 8×8 и 28×28 получаются аналогично, алгоритмы будут отличаться только параметрами линеаризации. Построенные модели являются основой для эксперимента, в ходе которого будут исследованы точность и скорость обработки в зависимости от размера входных данных и применяемой схемы ПГШ.

3.4 Экспериментальное исследование и сравнительный анализ полученных математических моделей

3.4.1 Порядок проведения экспериментального исследования

Для проведения экспериментального исследования на базе алгоритмов, представленных в Параграфе 3.1.1, были обучены 6 СНС учителей для изобра-

жений размером 8×8 и 6 СНС для изображений размером 28×28. Точность распознавания СНС находится в пределах 99-99.5%. Коэффициент сглаживания для дистилляции СНС учеников определен как $a = 0.875$. Далее были обучены СНС на базе математических моделей, полученных в Параграфе 3.1.2 с применением функций активации, рассмотренных в Параграфе 3.2, без дистилляции. Эти же модели СНС были обучены уже с применением дистилляции. После чего модели СНС-учеников, как с применением дистилляции, так и без нее, были квантованы и зашифрованы с помощью схемы ПГШ TFHE, для схемы CKKS квантизация не требуется.

Начнем экспериментальную часть с анализа функций активации ELU и ReLU, целью которого является определение степени полинома, дающей наилучшую точность распознавания.

Качество СНС оценивалось посредством следующих показателей:

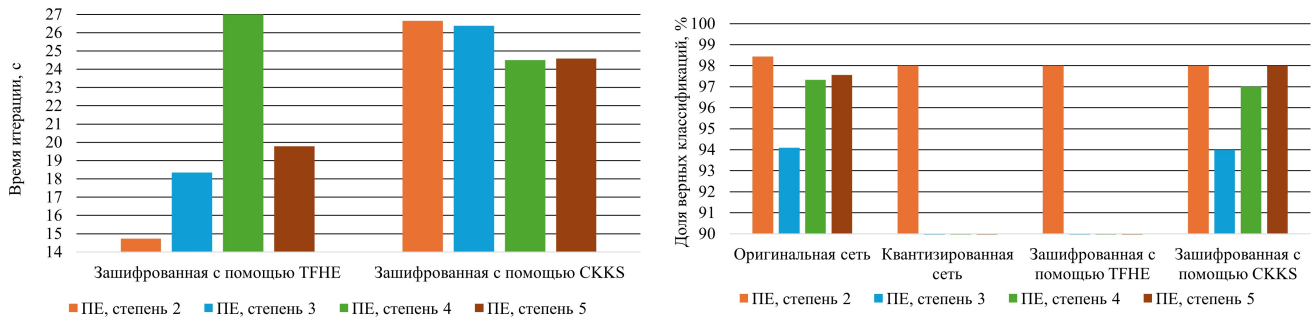
– производительность, которая определяется, как время, затраченное на одну итерацию работы СНС;
– точность распознавания тестовой подвыборки из 1000 изображений.

3.4.2 Определение приближающего полинома функции активации ELU с наивысшей точностью

Для приближенной функции активации ELU были получены 4 полинома различных степеней (3.5). Проанализируем СНСПЕСК с каждым из них в роли функции активации, как для изображений размером 8×8, так и для изображений размером 28×28, как с дистилляцией, так и без нее.

Проанализируем графики производительности СНСПЕСК и точности СНСПЕ и СНСПЕСК без дистилляции для изображений размером 8×8 (рис. 3.7).

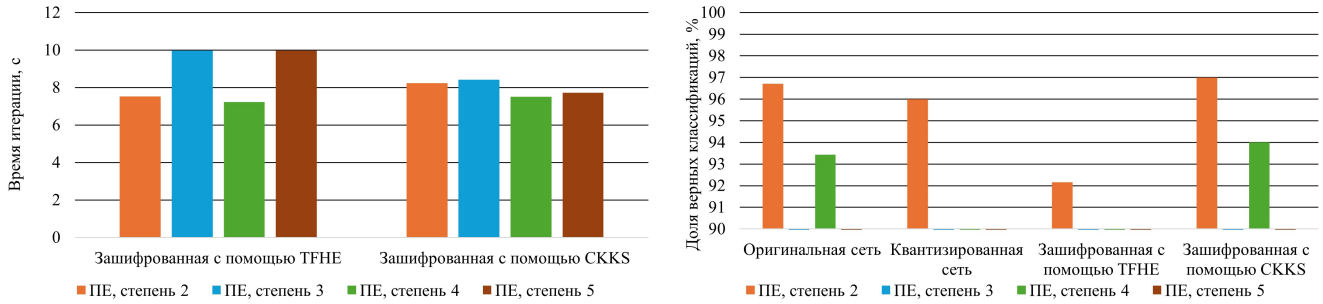
Анализируя графики, можно заметить, что наивысшая точность распознавания обеспечивается полиномом 2 степени независимо от схемы шифрования. С точки зрения производительности, для схемы TFHE полином 2 степени так же обладает и лучшей производительностью; для схемы CKKS полином 2 степени является наихудшим по производительности. Если же рассматривать выбранные критерии качества в совокупности, полином 2 степени наиболее



а) б)

Рисунок 3.7 — Результаты моделирования CHCПЕ(CK) без дистилляции для изображений размера 8×8: а) время работы; б) точность распознавания

предпочтителен как для схемы TFHE, так и для схемы CKKS. Теперь рассмотрим аналогичные графики для изображений размером 28×28 (рис. 3.8).



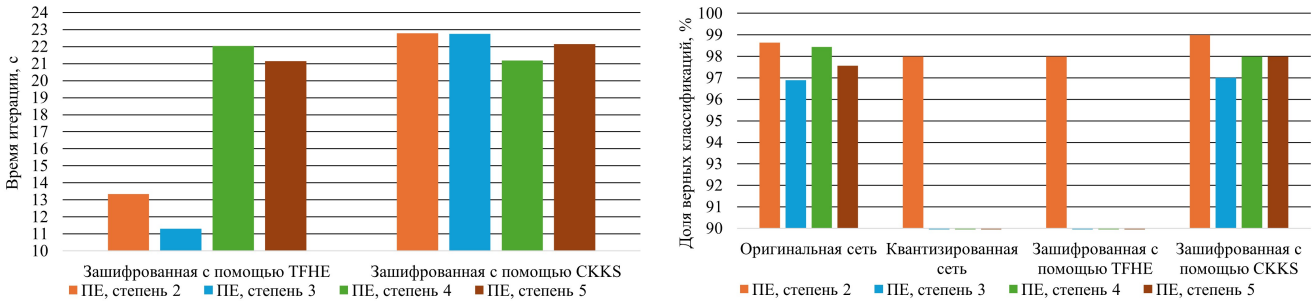
а) б)

Рисунок 3.8 — Результаты моделирования CHCПЕ(CK) без дистилляции для изображений размера 28×28: а) время работы; б) точность распознавания

Для изображений размера 28×28 полиномы 3 и 5 степеней не смогли обеспечить необходимую точность. Полином 4 степени смог обеспечить точность выше 93% для схемы CKKS, а полином 2 степени обеспечил точность выше 92% для схемы TFHE и выше 96% для схемы CKKS. С точки зрения производительности полином 2 степени проигрывает полиному 4 степени в пределах 0.5 секунды для схемы TFHE, и в пределах 1 секунды для схемы CKKS. Однако, выигрыш в точности полинома 2 степени над полиномом 4 степени нивелирует преимущество в производительности полинома 4 степени над полиномом 2 степени.

Теперь проанализируем результаты исследования CHCПЕCK с применением дистилляции. Основное отличие от сетей без дистилляции заключается в том, что сети с дистилляцией обладают большей точностью. Анализируя Рисунок 3.9, можно заметить, что точность для схемы CKKS достигает 99%, и в

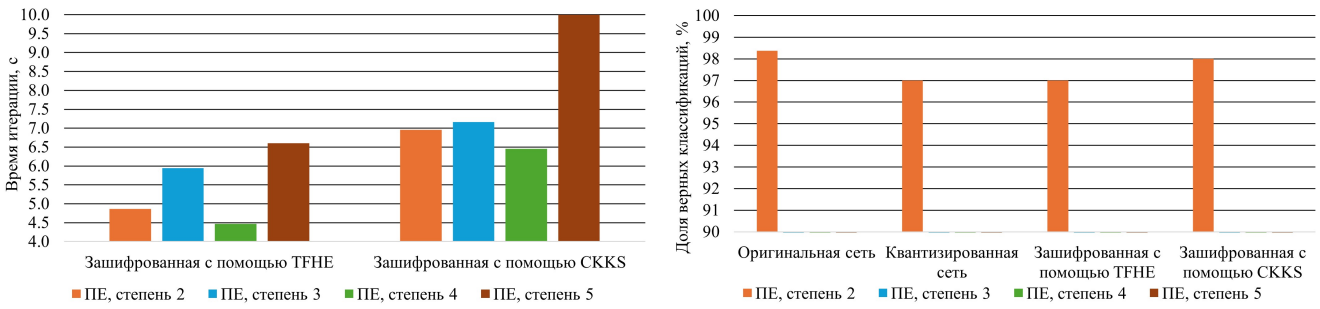
целом CHCПECK на базе схемы CKKS показали хорошую точность с полиномами 2, 4 и 5 степеней. Однако для CHCПECK на базе схемы TFHE удалось обеспечить высокую точность распознавания только при использовании полинома 2 степени. С точки зрения производительности ситуация практически идентичная ситуации для сетей без дистилляции, за тем исключением, что в целом у сетей с дистилляцией производительность выше.



а) б)

Рисунок 3.9 — Результаты моделирования CHCПE(CK) с дистилляцией для изображений размера 8×8: а) время работы; б) точность распознавания

Теперь рассмотрим результаты моделирования CHCПECK с дистилляцией для изображений размером 28×28 (рис. 3.10). В данном случае получить достаточную точность распознавания удалось только с использованием полинома 2 степени, как для схемы TFHE, так и для схемы CKKS. Для схемы CKKS полином 2 степени имеет более высокую точность, равную 98%, кроме того производительность полинома 2 степени для схемы CKKS немного ниже, чем у полинома 2 степени для TFHE.



а) б)

Рисунок 3.10 — Результаты моделирования CHCПE(CK) с дистилляцией для изображений размера 28×28: а) время работы; б) точность распознавания

Полученные результаты моделирования CHCПECK с полиномами разных степеней от 2 до 5 позволяют говорить о том, что наилучшим соотношением

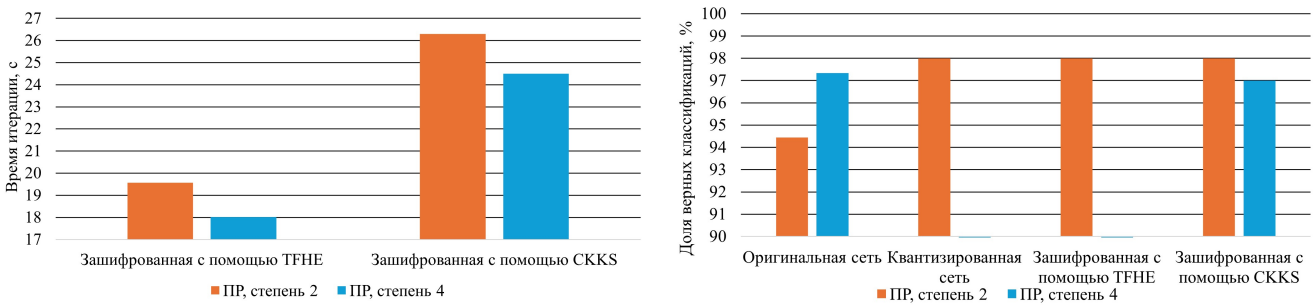
точности и производительности для схемы TFHE обладает полином 2 степени. В случае схемы CKKS можно утверждать, что полином 2 степени обеспечивает наилучшую точность при достаточной производительности.

Таким образом, для приближения функции активации ELU выбран полином 2 степени.

3.4.3 Определение приближающего полинома функции активации ReLU с наивысшей точностью

Теперь рассмотрим СНСПРСК. Для приближенной функции активации ReLU были получены полиномы 2 и 4 степеней (3.2). Аналогично СНСПЕСК было проведено моделирование СНСПРСК, как для изображений размера 8×8, так и для изображений размера 28×28, как с дистилляцией, так и без нее.

Анализируя производительность и точность СНСПРСК без дистилляции, можно заметить схожую с СНСПЕСК тенденцию, в данном случае лучшей точностью так же обладает полином 2 степени. Различия в производительности полиномов имеют схожее соотношение, полином 2 степени проигрывает в производительности полиному 4 степени (рис. 3.11).

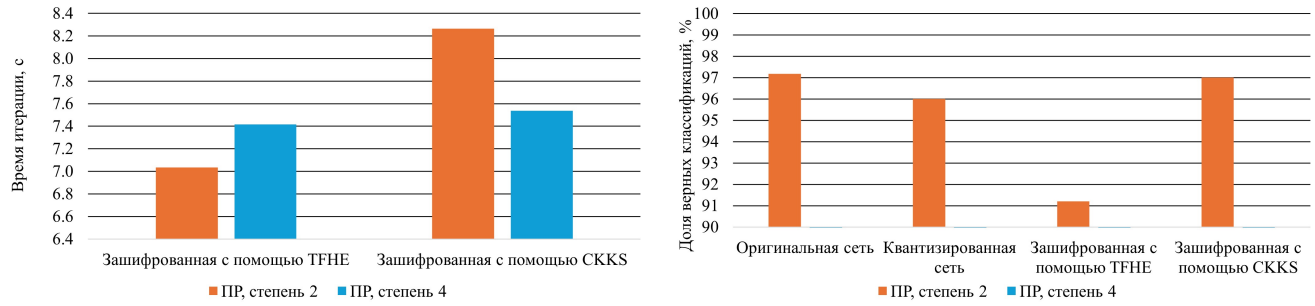


а) б)

Рисунок 3.11 — Результаты моделирования СНСПР(СК) без дистилляции для изображений размера 8×8: а) время работы; б) точность распознавания

В случае размера изображений 28×28 разрыв в точности СНСПРСК значительно выше (рис. 3.12). Так, например, точность распознавания для полинома 4 степени ниже 70% как для схемы TFHE, так и для схемы CKKS. Если анализировать производительность, то в случае CKKS производительность полинома 2 степени ниже в рамках 1 секунды. Для схемы TFHE, точность

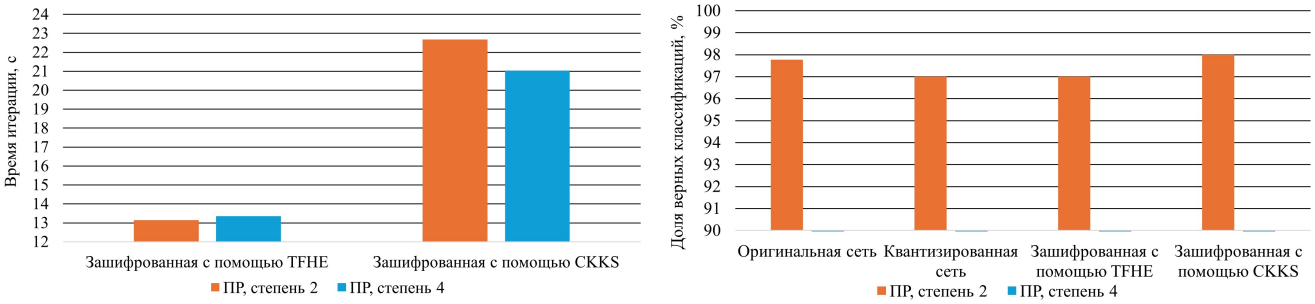
находится в окрестности 91% для полинома 2 степени. С точки зрения производительности, для TFHE наилучшие показатели так же у полинома 2 степени.



а) б)

Рисунок 3.12 — Результаты моделирования СНСПР(СК) без дистилляции для изображений размера 28 × 28: а) время работы; б) точность распознавания

Теперь рассмотрим СНСПРСК с дистилляцией. Проанализируем результаты моделирования для изображений размера 8 × 8 (рис. 3.13). В данном случае полином 2 степени так же показывает высокий результат по точности. СНСПРСК со схемой CKKS и полиномом 2 степени показывает точность 99%, а с полином 4 степени 98%. Если анализировать производительность для CKKS, то полином 4 степени превосходит полином 2 степени в рамках 2 секунд. Если анализировать СНСПРСК, основанную на схеме TFHE, то точность выше 90% обеспечивает только полином 2 степени, для него точность равна 98%.

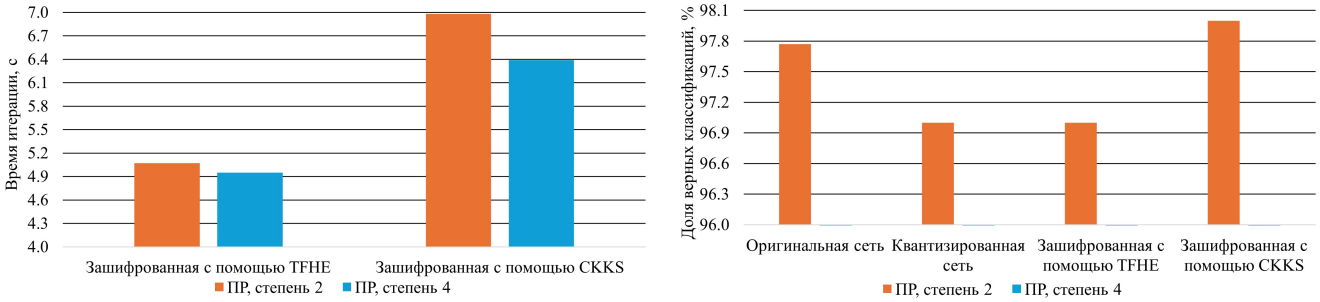


а) б)

Рисунок 3.13 — Результаты моделирования СНСПР(СК) с дистилляцией для изображений размера 8 × 8: а) время работы; б) точность распознавания

Теперь проанализируем СНСПРСК с дистилляцией для изображений размером 28 × 28 (рис. 3.14). В данном случае только полином 2 степени смог обеспечить точность выше 90%. СНСПРСК на базе схемы CKKS имеет точность распознавания 98%, а на базе схемы TFHE 97%. Производительность для полинома 2 степени хуже, чем для полинома 4 степени, однако совокупность

точности и производительности показывает явное преимущество полинома 2 степени.



а) б)

Рисунок 3.14 — Результаты моделирования СНСПР(СК) с дистилляцией для изображений размера 28×28: а) время работы; б) точность распознавания

На основе проведенного анализа можно заключить, что наивысшую точность обеспечивает полином 2 степени. Если рассматривать точность и производительность в совокупности, полином 2 степени так же наиболее предпочтителен.

Таким образом, для приближения функции активации ReLU выбран полином 2 степени.

3.4.4 Экспериментальное исследование полученных функций активации

Для приближения функций активации ReLU и ELU были выбраны полиномы 2 степени. Для функции активации ФАОК были обучены коэффициенты полинома так же 2 степени. Полиномы ФАОК для активации выходов сверточных слоев выглядят следующим образом:

$$y_{\text{ФАОК}_{8\times8}}(x) = -0.0588202 \cdot x^2 + 0.0128461 \cdot x + 0.0204959,$$
$$y_{\text{ФАОК}_{28\times28}}(x) = -0.003377 \cdot x^2 - 0.020059 \cdot x - 0.145558,$$
$$y_{\text{ФАОК\_}dist_{8\times8}}(x) = -0.058766 \cdot x^2 + 0.013047 \cdot x + 0.020398,$$
$$y_{\text{ФАОК\_}dist_{28\times28}}(x) = -0.002223 \cdot x^2 + 0.094038 \cdot x + 1.648167.$$

А для выходов скрытого слоя ФАОК имеют вид:

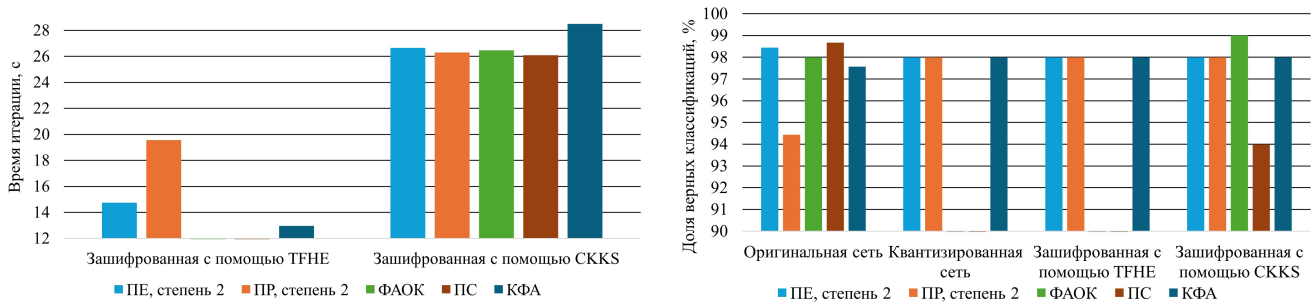
$$y_{\text{ФАОК}_{8\times8}}(x) = -0.0531935 \cdot x^2 - 0.0466886 \cdot x + 0.0349889,$$
$$y_{\text{ФАОК}_{28\times28}}(x) = -0.004618 \cdot x^2 + 0.033658 \cdot x + 3.277034,$$
$$y_{\text{ФАОК\_}dist_{8\times8}}(x) = -0.053145 \cdot x^2 - 0.046645 \cdot x + 0.034919,$$
$$y_{\text{ФАОК\_}dist_{28\times28}}(x) = 0.029810 \cdot x^2 - 0.089146 \cdot x - 2.771453.$$

Уточним, что приближенная сигмоидная функция активации определяется выражением (3.1), а квадратичная функция активации определяется, как квадраты значений выходов нейронов предыдущего слоя. Проанализируем и сравним результаты моделирования СНССК с рассмотренными функциями активации и выбранными полиномами для функций ELU, ReLU.

Для начала рассмотрим результаты моделирования СНССК для распознавания изображений размера 8×8 без дистилляции (рис. 3.15). Анализируя результаты точности распознавания изображений, можно заметить, что в открытом виде лучшую точность обеспечивает СНСПС с вещественными значениями, квантованные СНСПР, СНСПЕ и СНСКФА обеспечивают практически идентичную точность, а СНСФАОК и СНСПС после квантизации не смогли обеспечить точность выше 90%. СНССК со схемой TFHE демонстрируют лучшую точность равную 98% в случаях СНСПРСК, СНСПЕСК и СНСКФАСК. Для СНССК со схемой CKKS наилучшая точность достигается в случае СНСФАОКСК, и равна она 99%; худшая точность – 94% обеспечивается СНСПС. С точки зрения оценки производительности, при любой из реализаций схема TFHE показывает лучшие результаты. Учитывая не критичную разницу в точности СНСПРСК, СНСПЕСК и СНСКФАСК, оптимальнее использовать данные реализации со схемой TFHE. СНСПССК обеспечивает точность менее 90% со схемой TFHE, а со схемой CKKS обеспечивает наихудшую точность, равную 94%. СНСФАОКСК со схемой CKKS обеспечивает наилучшую точность и среднюю производительность.

СНССК для распознавания изображений размера 28×28 имеют следующие характеристики (рис. 3.16). Анализируя производительность СНССК, можно заметить схожую с результатами СНССК без дистилляции для изображений размера 8×8 тенденцию, а именно то, что СНССК на базе TFHE обладают лучшей производительностью по сравнению с СНССК на базе CKKS.

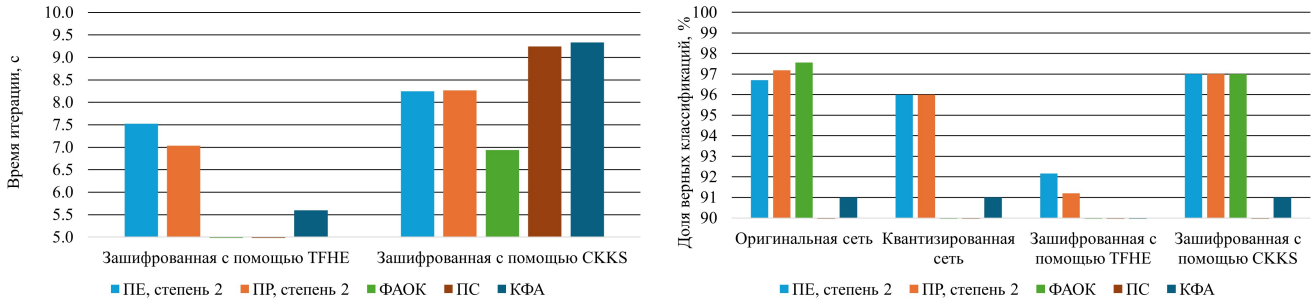
Однако, если анализировать точность распознавания, то СНССК на базе CKKS показывают лучшие результаты, СНСПРСК, СНСПЕСК и СПСФА-



а) б)

Рисунок 3.15 — Результаты моделирования СНС(СК) без дистилляции на базе полученных функций активации для изображений размера 8×8: а) время работы; б) точность распознавания

ОКСК демонстрируют точность распознавания на уровне 97%. Наилучшее сочетание точности распознавания и производительности достигается в случае СНСФАОКСК.

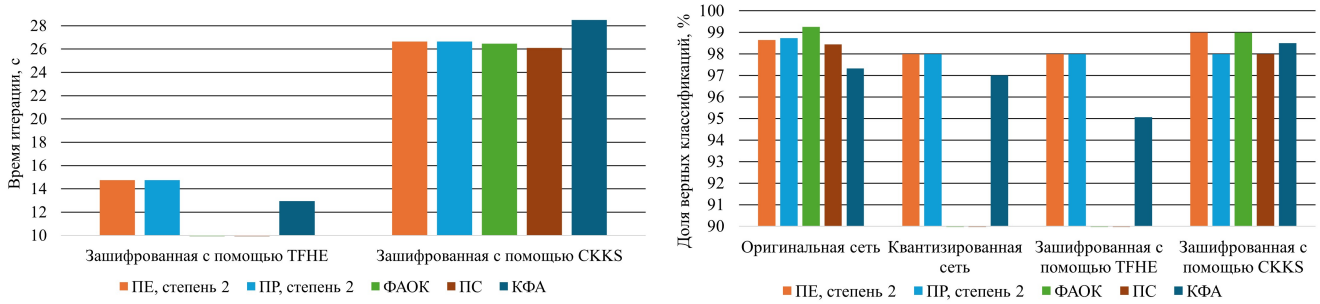


а) б)

Рисунок 3.16 — Результаты моделирования СНС(СК) без дистилляции на базе полученных функций активации для изображений размера 28×28: а) время работы; б) точность распознавания

Теперь проанализируем СНССК с применением дистилляции для изображений размером 8×8 (рис. 3.17). СНСПЕСК и СНСФАОКСК на базе CKKS показывают точность распознавания 99% и стабильную производительность. Если анализировать точность распознавания и производительность в совокупности, то лучший результат со схемой CKKS демонстрирует СНСФАОКСК. Если анализировать схему TFHE, то приемлемая точность распознавания достигается при использовании СНСПРСК и СНСПЕСК, и находится на уровне 98%, наилучшее сочетание точности и производительности достигается СНСПРСК.

Таким образом, анализируя полученные результаты можно заключить, что наилучшее сочетание точности распознавания и производительности достигается СНСПРСК на базе схемы TFHE, однако, стоит отметить, что наилучшая точность распознавания достигается при использовании реализаций СНССК со схемой CKKS.

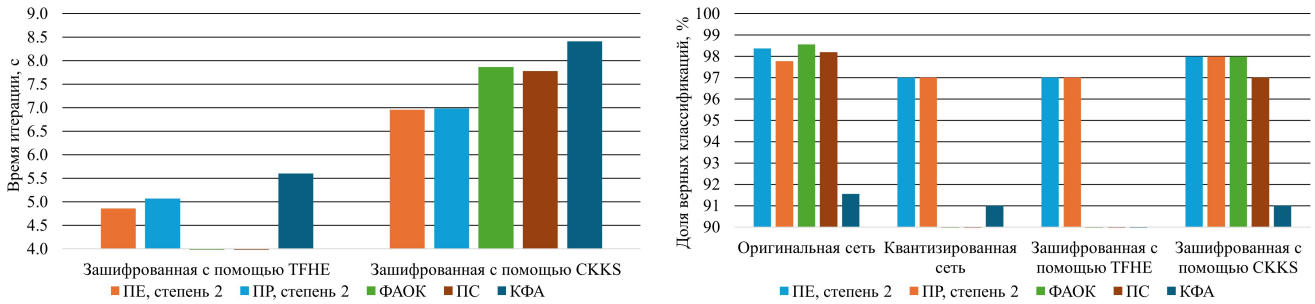


а) б)

Рисунок 3.17 — Результаты моделирования СНС(СК) с дистилляцией на базе полученных функций активации для изображений размера 8×8: а) время работы; б) точность распознавания

Исследование реализаций СНССК с применением дистилляции для распознавания изображений размером 28×28 показало следующие результаты (рис. 3.18). Наилучшую точность распознавания, примерно на уровне 98.5%, демонстрирует СНСФАОКСК на базе схемы CKKS. Другие СНССК на базе CKKS, кроме СНСФКАСК, показывают точность примерно равную 98%. СНСКФАСК демонстрирует точность распознавания на уровне 91.5%. Если анализировать СНССК на базе TFHE, то результаты исследования показывают, что только СНСПРСК и СНСПЕСК демонстрируют приемлемую точность, равную 97%. Анализируя точность распознавания и производительность в совокупности, можно говорить о том, что наиболее оптимальным будет применение СНСФАОКСК на базе схемы CKKS, с точки зрения точности и СНСПРСК на базе схемы TFHE с точки зрения производительности.

На основе проведенного экспериментального исследования можно заключить следующее. СНССК на базе CKKS характеризуются высокой точностью распознавания изображений, как размером 8×8, так и размером 28×28, и в большинстве случаев имеют наилучшее соотношение точности и производительности при распознавании изображений размером 28×28. СНССК на базе TFHE характеризуются высокой производительностью по сравнению с СНССК на базе CKKS и в большинстве случаев имеют наилучшее соотношение точности



а) б)

Рисунок 3.18 — Результаты моделирования CHC(CK) с дистилляцией на базе полученных функций активации для изображений размера 28×28: а) время работы; б) точность распознавания

и производительности при распознавании изображений размером 8×8. Кроме того, следует отметить, что далеко не для всех реализаций с различными приближенными функциями активации удалось добиться приемлемой точности распознавания, в особенности это касается реализаций со схемой TFHE.

Наблюдаемое различие в результатах работы CHCCK обусловлено тем, что сравниваемые схемы ПГШ работают на основе разных криптографических примитивов, в следствие чего опираются на разный математический аппарат.

Схема CKKS использует обычный гауссовский шум для сокрытия полезной информации, кроме того она использует полиномиальные приближения в кольце вычетов на базе чисел комплексного поля, что позволяет обрабатывать вещественные числа с заданной точностью. Схема имеет ограничение на количество разрешенных умножений, что сказывается на вычислительной сложности алгоритмов, использующих CKKS, поскольку для увеличения количества умножений необходимо применять бутстраппинг.

TFHE опирается на другую концепцию, в рамках которой шум определяется на основании математического ожидания и среднеквадратического отклонения конкретной области на поверхности тора. Кроме того, схема использует булеву арифметику, что позволяет применять LUT и операцию слепого вращения над ними. Однако, пространство тора не бесконечно и имеет ограничение на количество областей, если областей становится слишком много, то в таком случае математическое ожидание и среднеквадратическое отклонение могут определяться ошибочно, что приведет к некорректной работе схемы. Данную ситуацию можно увидеть на некоторых графиках, а именно при сравнении CHCCK без дистилляции и с применением дистилляции при обработке изобра-

жений размера 8×8 и 28×28; если партия обрабатываемых данных слишком большая для заданного тора, либо размер объекта партии больше поддерживаемого, то могут возникать ошибки и снижаться производительность расчетов, так как потребуется больше операций вращения.

Таким образом, получен набор моделей СНССК с различными функциями активации и различными схемами ПГШ, оптимальных для различных задач. Возвращаясь к РВС, рассмотренным в Параграфе 1.1, можно заключить, что для систем ТВ, таких как умные города и IoT, наиболее оптимальным будет применение СНССК на базе TFHE, а для систем ОВ применение СНССК на базе CKKS.

3.5 Программный комплекс проектирования сверточных нейронных сетей сохраняющих конфиденциальность

Главной задачей СНССК является обеспечение конфиденциальности данных пользователя в общедоступной среде (рис. 3.19). Для этого необходимо обучить СНС в открытом виде на наборе данных, который не является конфиденциальным (в рамках исследования все СНС были обучены на открытом наборе данных MNIST), так как задача конфиденциального обучения все еще является не решенной.

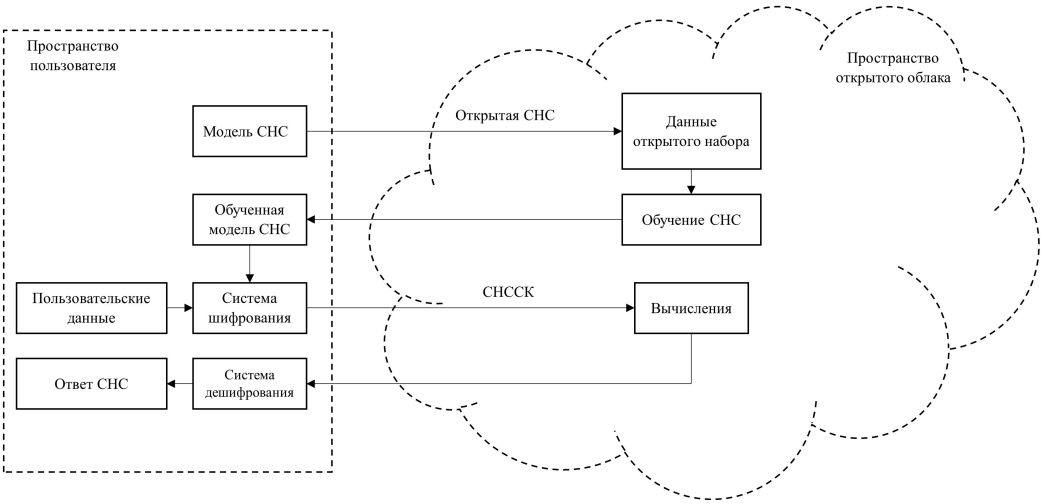


Рисунок 3.19 — Структурная схема работы СНССК

СНССК подразумевает оптимизацию самой СНС для применения в связке с шифрованием. Во 2 главе были разработаны методы оптимизации СНС для

применения с ПГШ и построены модели СНС с ПГШ для исследования предлагаемых методов. В Параграфах 3.1, 3.2 и 3.3 построенные модели были подробно проанализированы, обсуждение полученных на их основе экспериментальных результатов представлено в Параграфе 3.4. В ходе проведения исследований был разработан программный комплекс проектирования СНССК (далее ПКП). ПКП позволяет как исследовать, так и проектировать СНССК, включая их обучение, тестирование и измерение характеристик. ПКП так же включает в себя инструментарий для разработки СНССК [5] на базе представленных в работе функций активации, разработанных слоев и алгоритмов их взаимодействия [6]. Кроме того, ПКП содержит несколько демонстрационных моделей, которые были использованы в исследовании [4]. Подытоживая, ПКП содержит:

– разработанные функции активации;
– методы матричного умножения, сохраняющего конфиденциальность;
– алгоритмы сверточных слоев, сохраняющих конфиденциальность;
– алгоритмы слоев прямого прохода;
– алгоритмы обучения в коалиции с учителем (дистилляции).

ПКП использует такие библиотеки, как PyTorch, TorchVision, SkLearn, Concrete, Concrete-ML, SEAL, TenSEAL. Структурная схема ПКП представлена на Рисунке 3.20.

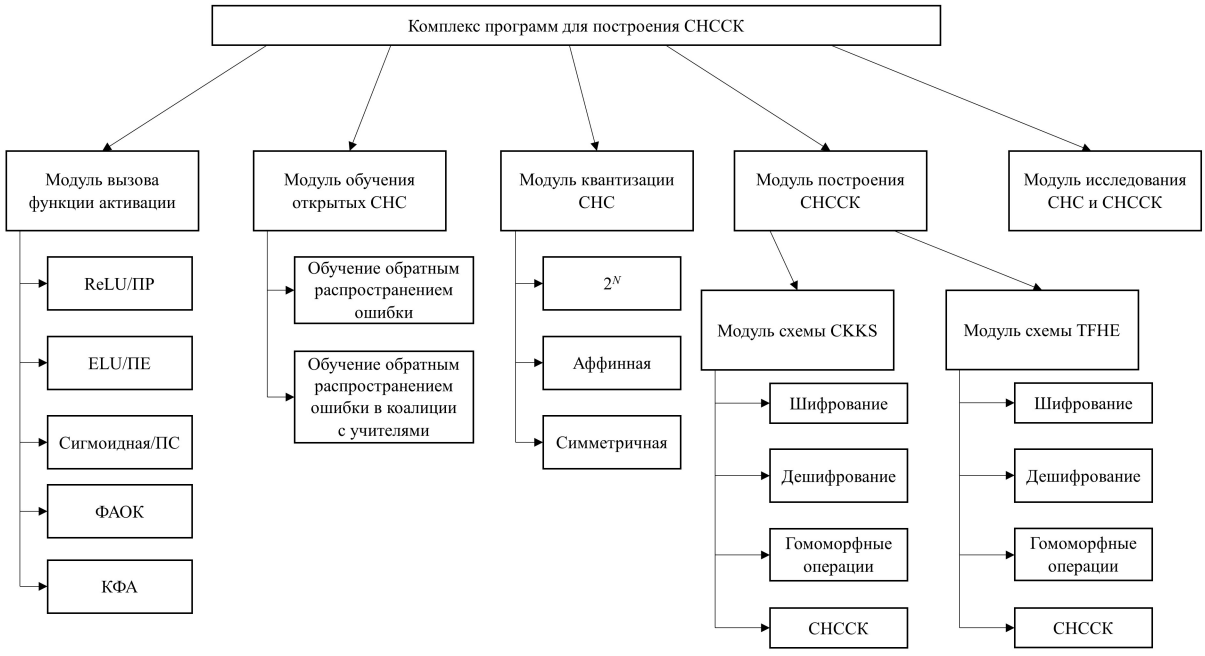


Рисунок 3.20 — Структурная схема работы ПКП

ПКП состоит из пяти основных модулей. Первый модуль отвечает за приближенные функции активации. Второй за обучение СНС в открытом виде.

Третий модуль используется для квантизации уже обученных СНС, и необходим для построения СНССК на базе схемы TFHE. Четвертый модуль отвечает за построение СНССК. Пятый модуль используется для исследования полученных СНС и СНССК. Рассмотрим подробнее каждый из представленных модулей.

3.5.1 Модуль приближенных функций активации

Основной функционал модуля заключается в реализации приближенных функций активации для их последующего использования в моделях СНС и СНССК. СНС или СНССК вызывает необходимую функцию активации, модуль вычисляет и возвращает либо ее значение, либо значение соответствующего приближающего полинома требуемой степени (если функция активации поддерживает более чем одну степень). Таким образом, на выходе получаем значения вызываемой функции активации $y(x)$, где x – значения на выходах слоя, после которого функция вызывается. Структурно данный модуль имеет следующий вид (рис. 3.21).

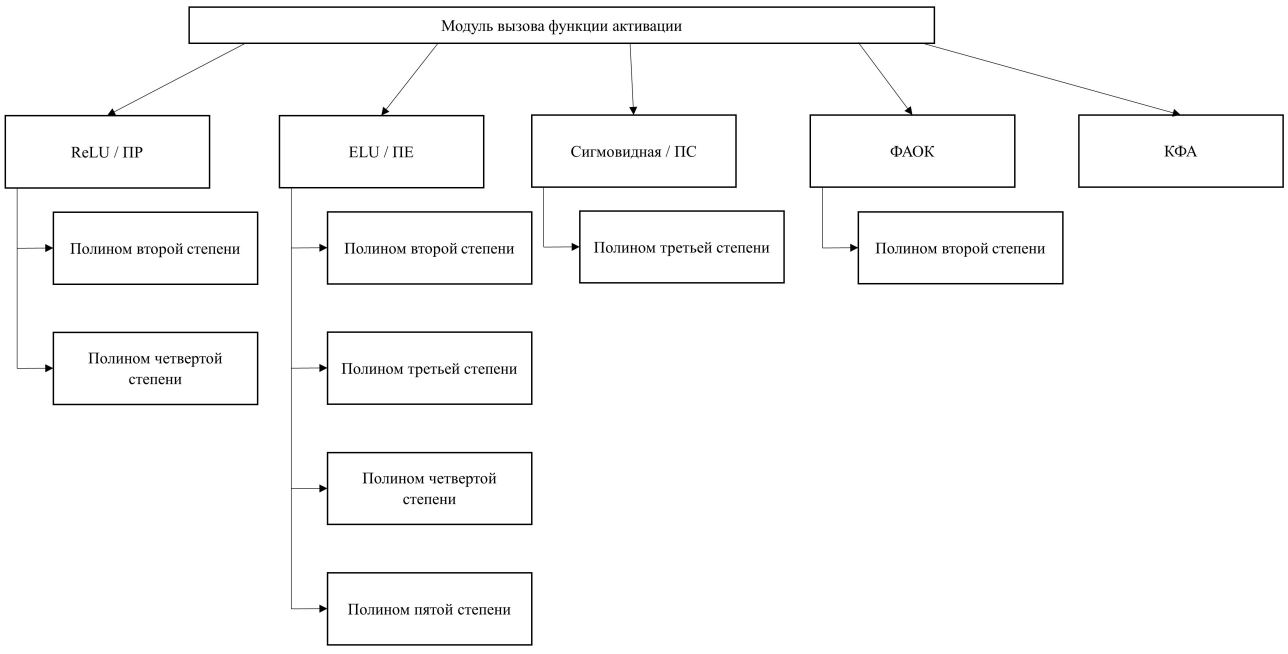


Рисунок 3.21 — Структура модуля приближенных функций активации

Модуль реализует Алгоритмы 3.6, 3.9, 3.12, 3.17. Стоит отметить, что функции активации ReLU и ELU заданные в явном виде вызываются из библио-

теки PyTorch. Квадратичная функция активации реализуется путем умножения выходов слоя самих на себя: в открытом виде это просто $x \cdot x$, в закрытом же виде реализуется вызовом функции возведения шифртекста в квадрат из библиотеки SEAL или Concrete в зависимости от схемы ПГШ.

В случае сигмоидной функции активации и функции активации ФАОК, может быть вызван только один полином, так как для сигмоиды в оригинальной работе [116] был получен полином третьей степени, а для ФАОК экспериментально было установлено, что полином второй степени дает конкурентоспособную точность при равном количестве эпох обучения по сравнению с другими функциями активации.

Модуль имеет связь с модулями обучения открытых СНС и построения СНССК, так как передает в них функции активации, представленные как в открытом, так и в закрытом виде.

3.5.2 Модуль обучения открытых сверточных нейронных сетей

Модуль реализует обучение открытых СНС алгоритмом обратного распространения ошибки как для случая с дистилляцией, так и без нее. В нем так же инициализируются модели открытых сетей, обучаются СНС учителей.

Модуль имеет связь с модулем приближенных функций активации. Полученные в результате обучения модели СНС передаются в модули построения СНССК, квантизации СНС и исследования СНС и СНССК. В первом случае на базе полученной после обучения модели СНС строится СНССК. Во втором случае полученная модель СНС квантуется. В третьем случае проводится исследование производительности и точности распознавания СНС.

3.5.3 Модуль квантизации сверточных нейронных сетей

Данный модуль применяется для квантизации СНС на базе Алгоритмов 2.3, 2.4 и 2.5. Квантизация необходима для работы схемы ПГШ TFHE, так как она не поддерживает обработку вещественных значений. Данный мо-

дуль имеет связь с модулем построения СНССК и модулем исследования СНС и СНССК. В первом случае квантованная СНС передается для последующего построения СНССК со схемой TFHE, во втором случае для исследования точности и производительности СНС.

3.5.4 Модуль построения сверточных нейронных сетей сохраняющих конфиденциальность

Данный модуль отвечает за построение СНССК. Его особенность заключается в том, что принимаемая от модуля обучения открытых СНС модель не изменяется, такие параметры слоев СНС как веса и смещения остаются открытыми. При работе СНССК зашифрованные входные данные проходят через слои СНС с открытыми параметрами и обрабатываются посредством Алгоритмов 2.8-2.10. Такой подход позволяет уменьшить избыточность и повысить скорость обработки, сохраняя конфиденциальность входных данных. Модуль поддерживает гомоморфные операции для двух схем ПГШ и позволяет проектировать модели СНССК на базе CKKS и TFHE.

Модуль имеет связь с модулем исследования СНС и СНССК для анализа точности и производительности.

3.5.5 Модуль исследования сверточных нейронных сетей открытых и сохраняющих конфиденциальность

Основной задачей модуля является тестирование СНС и СНССК на точность распознавания и производительность, которая определяется как время, затраченное на одну итерацию нейронной сети. Для получения статистически достоверного результата, как точность, так и производительность рассчитываются для всей тестовой подвыборки набора, т.е. выполняется 10000 итераций.

Модуль работает как с открытыми СНС с вещественными и целочисленными (квантованными) параметрами, так и с СНССК, использующими как схему ПГШ TFHE, так и схему CKKS.

3.6 Заключение по третьей главе

Данная глава посвящена разработке математических моделей СНССК. Разработаны математические модели открытых СНС на базе полученных приближенных функций активации. С учетом результатов исследования, представленного в Параграфе 2.3, подготовлены модели СНС учителей для реализации обучения с дистилляцией на двух наборах данных, а именно изображениях размера 8×8 и 28×28.

В ходе исследования моделей СНС и СНССК было установлено, что для приближенных функций активации ReLU и ELU наилучшее соотношение точности распознавания и производительности достигается при аппроксимации полиномами 2 степени. Кроме того исследование показало, что схема TFHE не может обеспечить точность распознавания больше 90% при использовании полиномиального приближения сигмоидной функции активации и функции активации ФАОК. Основным результатом исследования является экспериментально доказанное утверждение, что схема CKKS обеспечивает наилучшую точность распознавания изображений из обоих наборов данных, а схема TFHE обеспечивает наилучшую производительность при обработке обоих наборов данных. Если исследовать точность распознавания и производительность в совокупности, то в большинстве случаев схема CKKS обладает наилучшим соотношением при обработке изображений размером 28×28. Если рассматривать такую же совокупность при распознавании изображений размером 8×8, то здесь в большинстве случаев преимуществом обладает схема TFHE.

В ходе проведения исследований разработан ПКП СНССК, который позволяет дообучать разработанные в данном исследовании модели, а так же обучать новые, с использованием представленных в работе методов и инструментов: приближенных функций активации, квантизации, ПГШ на основе схем TFHE и CKKS. Так же ПКП содержит инструментарий для исследования СНС и СНССК, позволяющий оценить качество разработанных моделей.

Заключение

Работа посвящена исследованию методов и алгоритмов, необходимых для построения нейронных сетей, сохраняющих конфиденциальность с использованием ПГШ. Основным сдерживающим фактором для практического использования ПГШ для построения нейронных сетей, сохраняющих конфиденциальность является его высокая вычислительная сложность, складывающаяся из сложностей проблемных операций, таких как матричное умножение и определение знака закодированного числа, ограничений накладываемых параметрами шифрования. Основные полученные и представленные в работе результаты исследования можно сформулировать следующим образом:

1. Разработан метод матричного умножения, который повышает быстродействие за счет уменьшения количества гомоморфных операций при сохранении конфиденциальности данных пользователя, уменьшая вычислительную сложность алгоритма с $O(n^4)$ до $O(n^2)$.
2. Модифицирован метод дистилляции, который использует коалицию учителей для уменьшения размеров целевой сети, что влечет за собой сокращение потребления памяти примерно в 1500 раз, уменьшение времени обработки данных в среднем в 30 раз, при уменьшении доли верных классификаций от 0.5% до 1%.
3. Выполнена адаптация методов квантизации, которая позволяет применять целочисленные и логические схемы ПГШ для построения нейронных сетей, сохраняющих конфиденциальность с большей эффективностью с точки зрения потребления памяти более чем в 3.5 раза и уменьшение времени обработки данных на 20%.
4. Разработана ФАОК, позволяющая повысить долю верных классификаций в среднем на 1.5% по сравнению с функциями реализованными в библиотеках TenSEAL и Concete-ML.
5. Разработан комплекс программ и алгоритмов для проектирования и исследования приближенных функций активации позволяющий построить модель под конкретную задачу, повысить ее эффективность, а также расширить область применения для решения прикладных задач, требующих сохранения конфиденциальности.

Предложенная математическая модель, методы и алгоритмы позволяют не только расширить спектр задач, решаемых нейронными сетями на удаленных облачных сервисах, но и повысить эффективность решений с точки зрения вычислительной сложности и потребления памяти за счет оптимизации нейронных сетей и математического аппарата обработки конфиденциальных данных.

Обозначения и сокращения

ВМ	Виртуальная Машина
ГО	Глубокое Обучение
ИИ	Искусственный Интеллект
ИТ	Информационные Технологии
КФА	Квадратичная Функция Активации
МО	Машинное Обучение
НС	Нейронные Сети
ОВ	Облачные Вычисления
ООК	Обучение с Ошибками в Кольце
ООКТ	Обучение с Ошибками в Кольце и Торе
ООТ	Обучение с Ошибками в Торе
ОТ	Облачные Технологии
ПЕ	Приближенная функция активации ELU
ПКП	Программный Комплекс Проектирования СНССК
ПР	Приближенная функция активации ReLU
ПС	Приближенная Сигмоидная функция активации
ПО	Программное Обеспечение
РВС	Распределенные Вычислительные Системы
СНССК	Сверточная Нейронная Сеть, Сохраняющая Конфиденциальность
СНС	Сверточная Нейронная Сети
СНСКФА	Сверточная Нейронная Сеть с Квадратичной Функцией Активации
СНСПЕ	Сверточная Нейронная Сеть с Приближенной функцией активации ELU
СНСПЕСК	Сверточная Нейронная Сеть с Приближенной функцией активации ELU, Сохраняющая Конфиденциальность
СНСПР	Сверточная Нейронная Сеть с Приближенной функцией активации ReLU
СНСПРСК	Сверточная Нейронная Сеть с Приближенной функцией активации ReLU, Сохраняющая Конфиденциальность

СНСПС	Сверточная Нейронная Сеть с Приближенной Сигмоидной функцией активации
СНСФАОК	Сверточная Нейронная Сеть с Функцией Активации с Обучаемыми Коэффициентами
СНСПССК	Сверточная Нейронная Сеть с Приближенной Сигмоидной функцией активации, Сохраняющая Конфиденциальность
СНСФАОКСК	Сверточная Нейронная Сеть с Функцией Активации с Обучаемыми Коэффициентами, Сохраняющая Конфиденциальность
СРС	Схема Разделения Секрета
ТВ	Туманные вычисления
ТУ	Туманный Узел
УГ	Умный город
ФАОК	Функция Активации с Обучаемыми Коэффициентами
ЧГШ	Частично Гомоморфное Шифрование
ACI	Access Control Issues
AH	Account Hijacking
ANU	Abuse and Nefarious Use
API	Application Programming Interfaces
APT	Advanced Persistent Threats
BFV	Brakerski–Fan–Vercauteren
CKKS	Cheon–Kim–Kim–Song
DB	Data Breaches
DDoS	Distributed Denial of service
DL	Data Loss
DoS	Denial of Service
ELU	Exponential Linear Unit
GPT	Generative Pre-trained Transformer
IaaS	Infrastructure as a Service
IAs	Insecure APIs
IDD	Insufficient Due Diligence
IoT	Internet of Things
LAN	Local Area Network
LUT	Look Up Tablies
MI	Malicious Insider

MLaaS	Machine Learning as a Service
PaaS	Platform as a Service
ReLU	Rectified Linear Unit
SaaS	Software as a Service
SAV	System and Application Vulnerabilities
STI	Shared Technology Issues
TFHE	Fast Fully Homomorphic Encryption Over the Torus
WAN	Wide Area Network
WWW	World Wide Web

Список литературы

1. *Бабенко, М. Г.* Сравнительный анализ алгоритмов гомоморфного шифрования на основе обучения с ошибками [Текст] / М. Г. Бабенко, Е. И. Голимблевская, Е. М. Ширяев // Труды института системного программирования РАН. — 2020. — Т. 32, № 2. — С. 37—51.
2. *Глушков, В.* Вычислительная Система [Текст] / В. Глушков. — 1996.
3. *Голимблевская, Е. И.* Обзор программных реализаций методов гомоморфного шифрования [Текст] / Е. И. Голимблевская, Е. М. Ширяев // Фундаментальные проблемы управления производственными процессами в условиях перехода к индустрии 4.0. — 2020. — С. 284.
4. *Свидетельство о гос. регистрации программы для ЭВМ.* Программа моделирования сверточных нейронных сетей, сохраняющих конфиденциальность на туманных устройствах [Текст] / Е. М. Ширяев [и др.] ; Ф. государственное автономное образовательное учреждение высшего образования «Северо-Кавказский федеральный университет». — № 2025617506 ; заявл. 05.03.2025 ; опубл. 19.03.2025, 2025615824 (Рос. Федерация).
5. *Свидетельство о гос. регистрации программы для ЭВМ.* Программа построения сверточных нейронных сетей, сохраняющих конфиденциальность на туманных устройствах [Текст] / Е. М. Ширяев [и др.] ; Ф. государственное автономное образовательное учреждение высшего образования «Северо-Кавказский федеральный университет». — № 2025617046 ; заявл. 05.03.2025 ; опубл. 21.03.2025, 2025614617 (Рос. Федерация).
6. *Свидетельство о гос. регистрации программы для ЭВМ.* Программа, оптимизирующая использование нейронных сетей на туманных устройствах [Текст] / Е. М. Ширяев [и др.] ; Ф. государственное автономное образовательное учреждение высшего образования «Северо-Кавказский федеральный университет». — № 2025616724 ; заявл. 05.03.2025 ; опубл. 19.03.2025, 2025614357 (Рос. Федерация).

7. *Тынкевич, М. А.* Глава 7.6. 1. Интерполяционный многочлен Лагранжа [Текст] / М. А. Тынкевич // Численные методы анализа. — 2002.

8. *Ширяев, Е. М.* Разработка алгоритма конфиденциального поиска в распределенных системах [Текст] / Е. М. Ширяев, Н. Н. Кучеров, О. В. Криволапова // Современная наука и инновации. — 2023. — № 2. — С. 10—19.

9. *Юдицкий Д. И., А. И. Я. и.* Машинная арифметика в остаточных классах [Текст] / А. И. Я. и Юдицкий Д. И. — Сов. радио, 1968.

10. A survey on privacy-preserving machine learning with fully homomorphic encryption [Текст] / L. B. Pulido-Gaytan [и др.] // Latin American High Performance Computing Conference. — Springer. 2020. — С. 115—129.

11. An efficient method for comparing numbers and determining the sign of a number in RNS for even ranges [Текст] / A. Tchernykh [и др.] // Computation. — 2022. — Т. 10, № 2. — С. 17.

12. Data Storage with Increased Survivability and Reliability Based on the Residue Number System [Текст] / N. Kucherov [и др.] // Advances in Systems Science and Applications. — 2024. — Т. 24, № 02. — С. 166—186.

13. Homomorphic encryption for arithmetic of approximate numbers [Текст] / J. H. Cheon [и др.] // Advances in Cryptology–ASIACRYPT 2017: 23rd International Conference on the Theory and Applications of Cryptology and Information Security, Hong Kong, China, December 3-7, 2017, Proceedings, Part I 23. — Springer. 2017. — С. 409—437.

14. Improvement of the Approximate Method for the Comparison Operation in the RNS [Текст] / E. Shiryaev [и др.] // 2020 International Conference Engineering and Telecommunication (En&T). — IEEE. 2020. — С. 1—6.

15. RRNS base extension error-correcting code for performance optimization of scalable reliable distributed cloud data storage [Текст] / M. Babenko [и др.] // 2021 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW). — IEEE. 2021. — С. 548—553.

16. Weighted two-levels secret sharing scheme for multi-clouds data storage with increased reliability [Текст] / V. Miranda-López [и др.] // 2019 International Conference on High Performance Computing & Simulation (HPCS). — IEEE. 2019. — С. 915—922.

17. A Comparative Study of Secure Outsourced Matrix Multiplication Based on Homomorphic Encryption [Текст] / M. Babenko [et al.] // Big Data and Cognitive Computing. — 2023. — Vol. 7, no. 2. — P. 84.

18. A comparison of pooling methods for convolutional neural networks [Текст] / A. Zafar [et al.] // Applied Sciences. — 2022. — Vol. 12, no. 17. — P. 8643.

19. A Creative IoT agriculture platform for cloud fog computing [Текст] / T.-C. Hsu [et al.] // Sustainable Computing: Informatics and Systems. — 2020. — Vol. 28. — P. 100285.

20. A fog computing-based iot framework for precision agriculture [Текст] / E. Guardo [et al.] // Journal of Internet Technology. — 2018. — Vol. 19, no. 5. — P. 1401—1411.

21. A resources representation for resource allocation in fog computing networks [Текст] / A. Abouaomar [et al.]. — 2019.

22. A Review of Artificial Intelligence in Government and its Potential from a Public Policy Perspective [Текст] / D. Valle-Cruz [et al.] // Proceedings of the 20th Annual International Conference on Digital Government Research. — Dubai United Arab Emirates : ACM, 2019. — P. 91—99. — URL: https://dl.acm.org/doi/10.1145/3325112.3325242 (visited on 10/11/2023).

23. A survey of convolutional neural networks: analysis, applications, and prospects [Текст] / Z. Li [et al.] // IEEE transactions on neural networks and learning systems. — 2021. — Vol. 33, no. 12. — P. 6999—7019.

24. A survey on homomorphic encryption schemes: Theory and implementation [Текст] / A. Acar [et al.] // ACM Computing Surveys (Csur). — 2018. — Vol. 51, no. 4. — P. 1—35.

25. A survey on multi-cloud storage security: threats and countermeasures [Текст] / E. S. Bezuglova [et al.] // Computational Technologies. — 2023. — Vol. 28, no. 1. — P. 72—80.

26. A survey on resilience in the iot: Taxonomy, classification, and discussion of resilience mechanisms [Текст] / C. Berger [et al.] // ACM Computing Surveys (CSUR). — 2021. — Vol. 54, no. 7. — P. 1—39.

27. Adoption of artificial intelligence in banking services: an empirical analysis [Текст] / M. Rahman [et al.] // International Journal of Emerging Markets. — 2021. — URL: https://www.emerald.com/insight/content/doi/10.1108/IJOEM-06-2020-0724/full/html (visited on 10/11/2023) ; Publisher: Emerald Publishing Limited.

28. An Approach to Reducing Device Uncertainty in Fog-Cloud Computing [Текст] / N. Kucherov [et al.] // International Workshop on Advanced Information Security Management and Applications. — Springer. 2024. — P. 161—171.

29. An overview of fog computing and its security issues [Текст] / I. Stojmenovic [et al.] // Concurrency and Computation: Practice and Experience. — 2016. — Vol. 28, no. 10. — P. 2991—3005.

30. An overview of the BlueGene/L supercomputer [Текст] / N. R. Adiga [et al.] // SC'02: Proceedings of the 2002 ACM/IEEE Conference on Supercomputing. — IEEE. 2002. — P. 60—60.

31. Analytical Review of Confidential Artificial Intelligence: Methods and Algorithms for Deployment in Cloud Computing [Текст] / E. M. Shiriaev [et al.] // Programming and Computer Software. — 2024. — Vol. 50, no. 4. — P. 304—314.

32. *Anderson, D. P.* Boinc: A system for public-resource computing and storage [Текст] / D. P. Anderson // Fifth IEEE/ACM international workshop on grid computing. — IEEE. 2004. — P. 4—10.

33. Artificial intelligence in banking [Текст] / O. Kaya [et al.] // Artificial intelligence. — 2019. — (Visited on 10/11/2023).

34. ASimOV: A Framework for Simulation and Optimization of an Embedded AI Accelerator [Текст] / D. Hwang [et al.] // Micromachines. — 2021. — Vol. 12, no. 7. — P. 838. — URL: https://www.mdpi.com/2072-666X/12/7/838 (visited on 10/10/2023) ; Number: 7 Publisher: Multidisciplinary Digital Publishing Institute.

35. *Asmuth, C.* A modular approach to key safeguarding [Текст] / C. Asmuth, J. Bloom // IEEE transactions on information theory. — 1983. — Vol. 29, no. 2. — P. 208—210. — Publisher: IEEE.

36. *Barzu, M.* Compact sequences of co-primes and their applications to the security of CRT-based threshold schemes [Текст] / M. Barzu, F. L. Ţiplea, C. C. Drăgan // Information Sciences. — 2013. — Vol. 240. — P. 161—172. — Publisher: Elsevier.

37. *Beimel, A.* Secret-sharing schemes: A survey [Текст] / A. Beimel // International conference on coding and cryptology. — Springer. 2011. — P. 11—46.

38. *Bezuglova, E.* Development of an Approach to Confidential Learning with Errors in the Design of Neural Networks [Текст] / E. Bezuglova, E. Shiriaev // AISMA-2023: International Workshop on Advanced Information Security Management and Applications / ed. by Z. Raza [et al.]. — Cham : Springer Nature Switzerland, 2024. — P. 24—30.

39. *Bhardwaj, S.* Cloud computing: A study of infrastructure as a service (IAAS) [Текст] / S. Bhardwaj, L. Jain, S. Jain // International Journal of engineering and information Technology. — 2010. — Vol. 2, no. 1. — P. 60—63.

40. *Bisong, E.* Google Colaboratory [Текст] / E. Bisong // Building Machine Learning and Deep Learning Models on Google Cloud Platform. — Berkeley, CA : Apress, 2019. — P. 59—64. — URL: http://link.springer.com/10.1007/978-1-4842-4470-8_7 (visited on 10/11/2023).

41. *Blakley, G. R.* Rivest-Shamir-Adleman public key cryptosystems do not always conceal messages [Текст] / G. R. Blakley, I. Borosh // Computers & mathematics with applications. — 1979. — Vol. 5, no. 3. — P. 169—178.

42. *Bouvrie, J.* Notes on convolutional neural networks [Текст] / J. Bouvrie. — 2006. — URL: http://web.mit.edu/jvb/www/papers/cnn_tutorial.pdf (visited on 10/11/2023).

43. *Brakerski, Z.* (Leveled) Fully Homomorphic Encryption without Bootstrapping [Текст] / Z. Brakerski, C. Gentry, V. Vaikuntanathan // ACM Transactions on Computation Theory. — 2014. — Vol. 6, no. 3. — P. 1—36. — URL: https://dl.acm.org/doi/10.1145/2633600 (visited on 05/08/2024).

44. *Brown, T. B.* Language models are few-shot learners [Текст] / T. B. Brown // arXiv preprint arXiv:2005.14165. — 2020.

45. Building safe PaaS clouds: A survey on security in multitenant software platforms [Текст] / L. Rodero-Merino [et al.] // computers & security. — 2012. — Vol. 31, no. 1. — P. 96—108.

46. *Buske, D.* GIMPS Finds Another Prime! [Текст] / D. Buske, S. Keith // Math Horizons. — 2000. — Vol. 7, no. 4. — P. 19—21.

47. *Cadario, R.* Understanding, explaining, and utilizing medical artificial intelligence [Текст] / R. Cadario, C. Longoni, C. K. Morewedge // Nature human behaviour. — 2021. — Vol. 5, no. 12. — P. 1636—1642. — URL: https://www.nature.com/articles/s41562-021-01146-0 (visited on 10/11/2023) ; Publisher: Nature Publishing Group UK London.

48. *Carminati, M.* Impact and trends in embedding field programmable gate arrays and microcontrollers in scientific instrumentation [Текст] / M. Carminati, G. Scandurra // Review of Scientific Instruments. — 2021. — Vol. 92, no. 9. — URL: https://pubs.aip.org/aip/rsi/article-abstract/92/9/091501/1030652 (visited on 10/11/2023) ; Publisher: AIP Publishing.

49. Centralized fog computing security platform for IoT and cloud in healthcare system [Текст] / C. Thota [et al.] // Fog computing: Breakthroughs in research and practice. — IGI global, 2018. — P. 365—378.

50. *Chacón, I. B.* Ring Learning With Errors: A crossroads between postquantum cryptography, machine learning and number theory [Текст] / I. B. Chacón // arXiv preprint arXiv:1902.08551. — 2019.

51. *Chintala, S.* An overview of deep learning frameworks and an introduction to pytorch [Текст] / S. Chintala. — 2017. — URL: https://repository.gatech.edu/handle/1853/58786 (visited on 08/14/2024).

52. *Cloud, H.* The nist definition of cloud computing [Текст] / H. Cloud // National institute of science and technology, special publication. — 2011. — Vol. 800, no. 2011. — P. 145.

53. Cloud computing and grid computing 360-degree compared [Текст] / I. Foster [et al.] // 2008 grid computing environments workshop. — Ieee. 2008. — P. 1—10.

54. Comprehensive overview of backpropagation algorithm for digital image denoising [Текст] / A. Singh [et al.] // Electronics. — 2022. — Vol. 11, no. 10. — P. 1590.

55. *Computing, G.* Introduction to Grid Computing [Текст] / G. Computing. — 2002.

56. CONCRETE: Concrete operates on ciphertexts rapidly by extending TfhE [Текст] / I. Chillotti [et al.] // WAHC 2020-8th Workshop on Encrypted Computing & Applied Homomorphic Cryptography. — 2020.

57. Cryptonets: Applying neural networks to encrypted data with high throughput and accuracy [Текст] / R. Gilad-Bachrach [et al.] // International conference on machine learning. — PMLR, 2016. — P. 201—210. — URL: https://proceedings.mlr.press/v48/gilad-bachrach16.html (visited on 10/11/2023).

58. cuFHE [Текст]. — URL: https://github.com/vernamlab/cuFHE (visited on 11/10/2024) ; original-date: 2018-03-14T18:07:47Z.

59. *Cunningham, P.* Supervised Learning [Текст] / P. Cunningham, M. Cord, S. J. Delany // Machine Learning Techniques for Multimedia: Case Studies on Organization and Retrieval / ed. by M. Cord, P. Cunningham. — Berlin, Heidelberg : Springer Berlin Heidelberg, 2008. — P. 21—49. — URL: https://doi.org/10.1007/978-3-540-75171-7_2.

60. *Cusumano, M.* Cloud computing and SaaS as new computing platforms [Текст] / M. Cusumano // Communications of the ACM. — 2010. — Vol. 53, no. 4. — P. 27—29.

61. *Cybenko, G.* Approximation by superpositions of a sigmoidal function [Текст] / G. Cybenko // Mathematics of control, signals and systems. — 1989. — Vol. 2, no. 4. — P. 303—314.

62. *Davies, E. R.* Machine vision: theory, algorithms, practicalities [Текст] / E. R. Davies. — Elsevier, 2004.

63. *Deng, L.* Deep learning in natural language processing [Текст] / L. Deng. — Springer, 2018.

64. Distilling task-specific knowledge from bert into simple neural networks [Текст] / R. Tang [et al.] // arXiv preprint arXiv:1903.12136. — 2019.

65. *Dolui, K.* Comparison of edge computing implementations: Fog computing, cloudlet and mobile edge computing [Текст] / K. Dolui, S. K. Datta // 2017 Global Internet of Things Summit (GIoTS). — IEEE. 2017. — P. 1—6.

66. *Douligeris, C.* DDoS attacks and defense mechanisms: classification and state-of-the-art [Текст] / C. Douligeris, A. Mitrokotsa // Computer networks. — 2004. — Vol. 44, no. 5. — P. 643—666.

67. DT-RRNS: Routing protocol design for secure and reliable distributed smart sensors communication systems [Текст] / A. Gladkov [et al.] // Sensors. — 2023. — Vol. 23, no. 7. — P. 3738.

68. *Duan, J.* Privacy-preserving distributed deep learning based on secret sharing [Текст] / J. Duan, J. Zhou, Y. Li // Information Sciences. — 2020. — Vol. 527. — P. 108—127. — URL: https://www.sciencedirect.com/science/article/pii/S0020025520302553 (visited on 10/11/2023) ; Publisher: Elsevier.

69. *Dubey, S. R.* Activation functions in deep learning: A comprehensive survey and benchmark [Текст] / S. R. Dubey, S. K. Singh, B. B. Chaudhuri // Neurocomputing. — 2022. — Vol. 503. — P. 92—108.

70. Efficient implementation of the CKKS scheme using a quadratic residue number system [Текст] / E. M. Shiriaev [et al.] // 2021 IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering (ElConRus). — IEEE. 2021. — P. 665—669.

71. *ElGamal, T.* A public key cryptosystem and a signature scheme based on discrete logarithms [Текст] / T. ElGamal // IEEE transactions on information theory. — 1985. — Vol. 31, no. 4. — P. 469—472.

72. *Elsloo, T. van.* SEALion: a Framework for Neural Network Inference on Encrypted Data [Текст] / T. van Elsloo, G. Patrini, H. Ivey-Law. — 2019. — (Visited on 10/12/2023).

73. Enabling fog computing for industrial automation through time-sensitive networking (TSN) [Текст] / P. Pop [et al.] // IEEE Communications Standards Magazine. — 2018. — Vol. 2, no. 2. — P. 55—61.

74. *Falcetta, A.* Privacy-preserving deep learning with homomorphic encryption: An introduction [Текст] / A. Falcetta, M. Roveri // IEEE Computational Intelligence Magazine. — 2022. — Vol. 17, no. 3. — P. 14—25.

75. *Fan, J.* Somewhat practical fully homomorphic encryption [Текст] / J. Fan, F. Vercauteren // Cryptology ePrint Archive. — 2012.

76. *Finkelstein, R.* Analyzing Trend of Cloud Computing and it's Enablers using Gartner Strategic Technology [Текст] / R. Finkelstein. — 2004.

77. Fog computing approaches in IoT-enabled smart cities [Текст] / M. Songhorabadi [et al.] // Journal of Network and Computer Applications. — 2023. — Vol. 211. — P. 103557.

78. Fog computing for Healthcare 4.0 environment: Opportunities and challenges [Текст] / A. Kumari [et al.] // Computers & Electrical Engineering. — 2018. — Vol. 72. — P. 1—13.

79. Fog computing for sustainable smart cities: A survey [Текст] / C. Perera [et al.] // ACM Computing Surveys (CSUR). — 2017. — Vol. 50, no. 3. — P. 1—43.

80. Fog computing in healthcare–a review and discussion [Текст] / F. A. Kraemer [et al.] // IEEE Access. — 2017. — Vol. 5. — P. 9206—9222.

81. Fog computing: Platform and applications [Текст] / S. Yi [et al.] // 2015 Third IEEE workshop on hot topics in web systems and technologies (HotWeb). — IEEE. 2015. — P. 73—78.

82. Fully homomorphic encryption over the integers [Текст] / M. Van Dijk [et al.] // Advances in Cryptology–EUROCRYPT 2010: 29th Annual International Conference on the Theory and Applications of Cryptographic Techniques, French Riviera, May 30–June 3, 2010. Proceedings 29. — Springer. 2010. — P. 24—43.

83. General bootstrapping approach for RLWE-based homomorphic encryption [Текст] / A. Kim [et al.] // IEEE Transactions on Computers. — 2023.

84. *Gentry, C.* A fully homomorphic encryption scheme [Текст] / C. Gentry. — Stanford university, 2009.

85. *Gentry, C.* Homomorphic encryption from learning with errors: Conceptually-simpler, asymptotically-faster, attribute-based [Текст] / C. Gentry, A. Sahai, B. Waters // Advances in Cryptology–CRYPTO 2013: 33rd Annual Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2013. Proceedings, Part I. — Springer. 2013. — P. 75—92.

86. *Gold, E. M.* Language identification in the limit [Текст] / E. M. Gold // Information and control. — 1967. — Vol. 10, no. 5. — P. 447—474.

87. *Golimblevskaia, E.* Survey software implementations of homomorphic encryption methods [Текст] / E. Golimblevskaia, E. Shiriaev, N. Kucherov // Advances in Automation II: Proceedings of the International Russian Automation Conference, RusAutoConf2020, September 6-12, 2020, Sochi, Russia. — Springer. 2021. — P. 601—613.

88. *Goodfellow, I.* Deep learning [Текст] / I. Goodfellow, Y. Bengio, A. Courville. — MIT press, 2016. — URL: https://books.google.com/books?hl=ru&lr=&id=omivDQAAQBAJ&oi=fnd&pg=PR5&dq=Deep+Learning&ots=MNV5aolzSS&sig=waXAS6C-_v-48H2qbW9rMFkEhFY (visited on 10/11/2023).

89. google/fully-homomorphic-encryption [Текст]. — 2024. — URL: https://github.com/google/fully-homomorphic-encryption (visited on 08/14/2024) ; original-date: 2021-05-05T21:04:00Z.

90. Gpt-4 technical report [Текст] / J. Achiam [et al.] // arXiv preprint arXiv:2303.08774. — 2023.

91. H2O AI Cloud [Текст]. — URL: https://h2o.ai/platform/ai-cloud/ (visited on 10/11/2023).

92. *Halevi, S.* Algorithms in HElib [Текст] / S. Halevi, V. Shoup // Advances in Cryptology – CRYPTO 2014. Vol. 8616 / ed. by J. A. Garay, R. Gennaro. — Berlin, Heidelberg : Springer Berlin Heidelberg, 2014. — P. 554—571. — URL: http://link.springer.com/10.1007/978-3-662-44371-2_31 (visited on 06/14/2024) ; Series Title: Lecture Notes in Computer Science.

93. *Halevi, S.* Algorithms in helib [Текст] / S. Halevi, V. Shoup // Advances in Cryptology–CRYPTO 2014: 34th Annual Cryptology Conference, Santa Barbara, CA, USA, August 17-21, 2014, Proceedings, Part I 34. — Springer. 2014. — P. 554—571.

94. HEaaN [Текст]. — URL: https://heaan.it/ (visited on 04/02/2024).

95. *Herrick, D. R.* Google this! Using Google apps for collaboration and productivity [Текст] / D. R. Herrick // Proceedings of the 37th annual ACM SIGUCCS fall conference: communication and collaboration. — 2009. — P. 55—64.

96. High-Speed Convolution Core Architecture for Privacy-Preserving Neural Networks [Текст] / M. A. Lapina [et al.] // Programming and Computer Software. — 2024. — Vol. 50, no. 6. — P. 417—424.

97. Homomorphic Encryption Standardization – An Open Industry / Government / Academic Consortium to Advance Secure Computation [Текст]. — URL: https://homomorphicencryption.org/ (visited on 04/01/2024).

98. *Huang, X.* Deep-learning based climate downscaling using the super-resolution method [Текст] / X. Huang. — 1981.

99. *huelse.* Huelse/SEAL-Python [Текст] / huelse. — 2024. — URL: https://github.com/Huelse/SEAL-Python (visited on 11/10/2024) ; original-date: 2019-07-15T09:16:52Z.

100. *Ibarrondo, A.* Pyfhel: PYthon For Homomorphic Encryption Libraries [Текст] / A. Ibarrondo, A. Viand // Proceedings of the 9th on Workshop on Encrypted Computing & Applied Homomorphic Cryptography. — Virtual Event Republic of Korea : ACM, 2021. — P. 11—16. — URL: https://dl.acm.org/doi/10.1145/3474366.3486923 (visited on 10/15/2024).

101. Implementation and performance evaluation of RNS variants of the BFV homomorphic encryption scheme [Текст] / A. Al Badawi [et al.] // IEEE Transactions on Emerging Topics in Computing. — 2019. — Vol. 9, no. 2. — P. 941—956.

102. Improved programmable bootstrapping with larger precision and efficient arithmetic circuits for TFHE [Текст] / I. Chillotti [et al.] // Advances in Cryptology–ASIACRYPT 2021: 27th International Conference on the Theory and Application of Cryptology and Information Security, Singapore, December 6–10, 2021, Proceedings, Part III 27. — Springer. 2021. — P. 670—699.

103. *Kaelbling, L. P.* Reinforcement learning: A survey [Текст] / L. P. Kaelbling, M. L. Littman, A. W. Moore // Journal of artificial intelligence research. — 1996. — Vol. 4. — P. 237—285.

104. *Khan, S.* Fog computing security: a review of current applications and security solutions [Текст] / S. Khan, S. Parkinson, Y. Qin // Journal of Cloud Computing. — 2017. — Vol. 6. — P. 1—22.

105. *Khare, S.* Literature Review on Application of Artificial Neural Network (Ann) In Operation of Reservoirs [Текст] / S. Khare, A. Gajbhiye // International Journal of computational Engineering research (IJCER) IJCER| June 2013| VOL 3 ISSUE 6. — 1943. — P. 63.

106. *Kim, K.-H.* Symmetry quantification and mapping using convergent beam electron diffraction [Текст] / K.-H. Kim, J.-M. Zuo // Ultramicroscopy. — 2013. — Vol. 124. — P. 71—76.

107. *Klas, G. I.* Fog computing and mobile edge cloud gain momentum open fog consortium, etsi mec and cloudlets [Текст] / G. I. Klas // Google Scholar. — 2015. — Vol. 1, no. 1. — P. 1—13.

108. *Klonoff, D. C.* Fog computing and edge computing architectures for processing data from diabetes devices connected to the medical internet of things [Текст] / D. C. Klonoff // Journal of diabetes science and technology. — 2017. — Vol. 11, no. 4. — P. 647—652.

109. *Klonoff, D. C.* Fog computing and edge computing architectures for processing data from diabetes devices connected to the medical internet of things [Текст] / D. C. Klonoff // Journal of diabetes science and technology. — 2017. — Vol. 11, no. 4. — P. 647—652. — Publisher: SAGE Publications Sage CA: Los Angeles, CA.

110. *Kocabas, O.* Utilizing homomorphic encryption to implement secure and private medical cloud computing [Текст] / O. Kocabas, T. Soyata // 2015 IEEE 8th International Conference on Cloud Computing. — IEEE, 2015. — P. 540—547.

111. *Kratzke, N.* A brief history of cloud application architectures [Текст] / N. Kratzke // Applied Sciences. — 2018. — Vol. 8, no. 8. — P. 1368.

112. *Kucherov, N.* On the Way to Building Reliable and Secure Cloud-Based Data Processing Systems [Текст] / N. Kucherov, E. Shiriaev, E. Bezuglova // International Workshop on Advanced Information Security Management and Applications. — Springer. 2024. — P. 152—160.

113. Lattigo: lattice-based multiparty homomorphic encryption library in Go [Текст]. — URL: https://github.com/tuneinsight/lattigo (visited on 11/10/2024) ; original-date: 2019-08-12T15:14:22Z.

114. *Lehner, W.* Database as a service (DBaaS) [Текст] / W. Lehner, K.-U. Sattler // 2010 IEEE 26th International Conference on Data Engineering (ICDE 2010). — IEEE. 2010. — P. 1216—1217.

115. *Liu, R.* A review of medical artificial intelligence [Текст] / R. Liu, Y. Rong, Z. Peng // Global Health Journal. — 2020. — Vol. 4, no. 2. — P. 42—45. — URL: https://www.sciencedirect.com/science/article/pii/S2414644720300208 (visited on 10/11/2023) ; Publisher: Elsevier.

116. Logistic regression over encrypted data from fully homomorphic encryption [Текст] / H. Chen [et al.] // BMC medical genomics. — 2018. — Vol. 11. — P. 3—12.

117. *Mahesh, B.* Machine learning algorithms-a review [Текст] / B. Mahesh // International Journal of Science and Research (IJSR).[Internet]. — 2020. — Vol. 9, no. 1. — P. 381—386.

118. *Manvi, S. S.* Resource management for Infrastructure as a Service (IaaS) in cloud computing: A survey [Текст] / S. S. Manvi, G. K. Shyam // Journal of network and computer applications. — 2014. — Vol. 41. — P. 424—440.

119. *Mellit, A.* MPPT-based artificial intelligence techniques for photovoltaic systems and its implementation into field programmable gate array chips: Review of current status and future perspectives [Текст] / A. Mellit, S. A. Kalogirou // Energy. — 2014. — Vol. 70. — P. 1—21. — URL: https://www.sciencedirect.com/science/article/pii/S0360544214003661 (visited on 10/11/2023) ; Publisher: Elsevier.

120. *Meng, S.* Enhanced monitoring-as-a-service for effective cloud management [Текст] / S. Meng, L. Liu // IEEE Transactions on Computers. — 2012. — Vol. 62, no. 9. — P. 1705—1720.

121. Microsoft SEAL [Текст]. — 2023. — URL: https://github.com/microsoft/SEAL (visited on 10/12/2023).

122. *Mignotte, M.* How to share a secret [Текст] / M. Mignotte // Workshop on cryptography. — Springer, 1982. — P. 371—375.

123. *Mishra, A.* Artificial Intelligence Accelerators [Текст] / A. Mishra, P. Yadav, S. Kim // Artificial Intelligence and Hardware Accelerators / ed. by A. Mishra [et al.]. — Cham : Springer International Publishing, 2023. — P. 1—52. — URL: https://doi.org/10.1007/978-3-031-22170-5_1.

124. *Mishra, D.* Artificial Intelligence Candidate Recruitment System using Software as a Service (SaaS) Architecture [Текст] / D. Mishra, S. Shekhar //. — 2018. — URL: https://api.semanticscholar.org/CorpusID:231796885.

125. *Mishra, S.* AI business model: an integrative business approach [Текст] / S. Mishra, A. R. Tripathi // Journal of Innovation and Entrepreneurship. — 2021. — Vol. 10, no. 1. — P. 18. — URL: https://innovation-entrepreneurship.springeropen.com/articles/10.1186/s13731-021-00157-5 (visited on 10/11/2023).

126. MLaaS in the wild: Workload analysis and scheduling in Large-Scale heterogeneous GPU clusters [Текст] / Q. Weng [et al.] // 19th USENIX Symposium on Networked Systems Design and Implementation (NSDI 22). — 2022. — P. 945—960. — URL: https://www.usenix.org/conference/nsdi22/presentation/weng (visited on 10/11/2023).

127. MNIST handwritten digit database, Yann LeCun, Corinna Cortes and Chris Burges [Текст]. — URL: http://yann.lecun.com/exdb/mnist/ (visited on 10/12/2023).

128. Modeling Hyperchaotic Datasets for Neural Networks [Текст] / E. Shiriaev [et al.] // International Conference on Mathematics and its Applications in new Computer Systems. — Springer. 2021. — P. 441—453.

129. Modification and Adaptation of Methods and Algorithms of the Active Security Concept for Fog Systems [Текст] / E. Shiriaev [et al.] // International Workshop on Advanced Information Security Management and Applications. — Springer. 2024. — P. 277—285.

130. *Needham, R. M.* The Cambridge distributed computing system [Текст] / R. M. Needham, A. J. Herbert. — 1983.

131. Neural network method for base extension in residue number system [Текст] / M. Babenko [et al.] // CEUR Workshop Proceedings. — 2020. — P. 9—22.

132. NVIDIA NGC | NVIDIA [Текст]. — URL: https://www.nvidia.com/en-us/gpu-cloud/ (visited on 10/11/2023).

133. OpenFHE.org – OpenFHE – Open-Source Fully Homomorphic Encryption Library [Текст]. — URL: https://www.openfhe.org/ (visited on 04/01/2024).

134. Optimized relinearization algorithm of the multikey homomorphic encryption scheme [Текст] / X. Yang [et al.] // Tsinghua Science and Technology. — 2021. — Vol. 27, no. 3. — P. 642—652.

135. *Park, H.* Chapter Three - Hardware accelerator systems for artificial intelligence and machine learning [Текст] / H. Park, S. Kim // Hardware Accelerator Systems for Artificial Intelligence and Machine Learning. Vol. 122 / ed. by S. Kim, G. C. Deka. — Elsevier, 2021. — P. 51—95. — (Advances in Computers). — URL: https://www.sciencedirect.com/science/article/pii/S0065245820300929.

136. *Peter, N.* Fog computing and its real time applications [Текст] / N. Peter // Int. J. Emerg. Technol. Adv. Eng. — 2015. — Vol. 5, no. 6. — P. 266—269.

137. *Pitts, W.* The linear theory of neuron networks: The dynamic problem [Текст] / W. Pitts // The bulletin of mathematical biophysics. — 1943. — Vol. 5. — P. 23—31.

138. Practical Two-party Privacy-preserving Neural Network Based on Secret Sharing [Текст] / Z. Ge [et al.]. — 2021. — URL: http://arxiv.org/abs/2104.04709 (visited on 10/11/2023) ; arXiv:2104.04709 [cs].

139. Privacy-Preserving Classification on Deep Neural Network [Текст] / H. Chabanne [et al.]. — 2017. — URL: https://eprint.iacr.org/2017/035 (visited on 08/11/2024) ; Publication info: Preprint. MINOR revision.

140. Privacy-Preserving Generative Deep Neural Networks Support Clinical Data Sharing [Текст] / B. K. Beaulieu-Jones [et al.] // Circulation: Cardiovascular Quality and Outcomes. — 2019. — Vol. 12, no. 7. — e005122. — URL: https://www.ahajournals.org/doi/10.1161/CIRCOUTCOMES.118.005122 (visited on 10/11/2023).

141. Privacy-preserving machine learning with fully homomorphic encryption for deep neural network [Текст] / J.-W. Lee [et al.] // IEEE Access. — 2022. — Vol. 10. — P. 30039—30054. — URL: https://ieeexplore.ieee.org/abstract/document/9734024/ (visited on 10/12/2023) ; Publisher: IEEE.

142. Quantification of information transfer via cellular signal transduction pathways [Текст] / B. N. Kholodenko [et al.] // FEBS letters. — 1997. — Vol. 414, no. 2. — P. 430—434.

143. *Radford, A.* Improving language understanding by generative pre-training [Текст] / A. Radford. — 2018.

144. *Rani, S.* Fog computing in industry 4.0: Applications and challenges—A research roadmap [Текст] / S. Rani, A. Kataria, M. Chauhan // Energy conservation solutions for fog-edge computing paradigms. — 2022. — P. 173—190.

145. *Rawat, W.* Deep convolutional neural networks for image classification: A comprehensive review [Текст] / W. Rawat, Z. Wang // Neural computation. — 2017. — Vol. 29, no. 9. — P. 2352—2449. — URL: https://ieeexplore.ieee.org/abstract/document/8016501/ (visited on 10/11/2023) ; Publisher: MIT Press.

146. *Reis, J.* Artificial Intelligence in Government Services: A Systematic Literature Review [Текст] / J. Reis, P. E. Santo, N. Melão // New Knowledge in Information Systems and Technologies. Vol. 930 / ed. by Á. Rocha [et al.]. — Cham : Springer International Publishing, 2019. — P. 241—252. — URL: http://link.springer.com/10.1007/978-3-030-16181-1_23 (visited on 10/11/2023) ; Series Title: Advances in Intelligent Systems and Computing.

147. Reliablity and Security for Fog Computing Systems [Текст] / E. Shiriaev [et al.] // Information. — 2024. — Vol. 15, no. 6. — P. 317.

148. Robust and privacy-preserving decentralized deep federated learning training: Focusing on digital healthcare applications [Текст] / Y. Tian [et al.] // IEEE/ACM Transactions on Computational Biology and Bioinformatics. — 2023. — URL: https://ieeexplore.ieee.org/abstract/document/10058838/ (visited on 01/18/2024) ; Publisher: IEEE.

149. *Ruiz-Rosero, J.* Field programmable gate array applications–A scientometric review [Текст] / J. Ruiz-Rosero, G. Ramirez-Gonzalez, R. Khanna // Computation. — 2019. — Vol. 7, no. 4. — P. 63. — URL: https://www.mdpi.com/2079-3197/7/4/63 (visited on 10/11/2023) ; Publisher: MDPI.

150. *Sabireen, H.* A review on fog computing: Architecture, fog with IoT, algorithms and research challenges [Текст] / H. Sabireen, V. Neelanarayanan // Ict Express. — 2021. — Vol. 7, no. 2. — P. 162—176.

151. *Sadok, H.* Artificial intelligence and bank credit analysis: A review [Текст] / H. Sadok, F. Sakka, M. E. H. El Maknouzi // Cogent Economics & Finance. — 2022. — Vol. 10, no. 1. — P. 2023262. — URL: https://www.tandfonline.com/doi/full/10.1080/23322039.2021.2023262 (visited on 10/11/2023).

152. *Samuel, A. L.* Machine learning [Текст] / A. L. Samuel // The Technology Review. — 1959. — Vol. 62, no. 1. — P. 42—45.

153. *Sarkar, S.* Assessment of the suitability of fog computing in the context of internet of things [Текст] / S. Sarkar, S. Chatterjee, S. Misra // IEEE Transactions on Cloud Computing. — 2015. — Vol. 6, no. 1. — P. 46—59.

154. Secure logistic regression based on homomorphic encryption: Design and evaluation [Текст] / M. Kim [et al.] // JMIR medical informatics. — 2018. — Vol. 6, no. 2. — e8805. — Publisher: JMIR Publications Inc., Toronto, Canada.

155. Secure outsourced matrix computation and application to neural networks [Текст] / X. Jiang [et al.] // Proceedings of the 2018 ACM SIGSAC conference on computer and communications security. — 2018. — P. 1209—1222.

156. *Seesing, A.* Evotest: Test case generation using genetic programming and software analysis [Текст] / A. Seesing // Operations Research. — 1954. — Vol. 2. — P. 393—410.

157. *Shamir, A.* How to share a secret [Текст] / A. Shamir // Communications of the ACM. — 1979. — Vol. 22, no. 11. — P. 612—613. — Publisher: ACm New York, NY, USA.

158. *Shawash, J.* Real-time nonlinear parameter estimation using the Levenberg–-Marquardt algorithm on field programmable gate arrays [Текст] / J. Shawash, D. R. Selviah // IEEE Transactions on industrial electronics. — 2012. — Vol. 60, no. 1. — P. 170—176. — URL: https://ieeexplore.ieee.org/abstract/document/6129410/ (visited on 10/11/2023) ; Publisher: IEEE.

159. *Shiriaev, E. M.* One Plaintext Attack on the BFV Scheme [Текст] / E. M. Shiriaev, A. S. Nazarov, N. A. Sotikova // 2021 IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering (ElConRus). — IEEE. 2021. — P. 670—673.

160. *Shiriaev, E.* Load Balancing Methods for Distributed Data Storage: Challenges and Opportunities [Текст] / E. Shiriaev // Current Problems of Applied Mathematics and Computer Systems / ed. by A. Alikhanov [et al.]. — Cham : Springer Nature Switzerland, 2024. — P. 95—104.

161. *Shiriaev, E.* Analytical Review of Cryptographic Primitives to Use in Fog Computing [Текст] / E. Shiriaev, N. Kucherov, E. Bezuglova // International Workshop on Advanced Information Security Management and Applications. — Springer. 2024. — P. 267—276.

162. *Shokri, R.* Privacy-Preserving Deep Learning [Текст] / R. Shokri, V. Shmatikov // Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security. — Denver Colorado USA : ACM, 2015. — P. 1310—1321. — URL: https://dl.acm.org/doi/10.1145/2810103.2813687 (visited on 10/11/2023).

163. *Singh, K.* A survey of static scheduling algorithm for distributed computing system [Текст] / K. Singh, M. Alam, S. K. Sharma // International Journal of Computer Applications. — 2015. — Vol. 129, no. 2. — P. 25—30.

164. *Smarr, L.* Metacomputing [Текст] / L. Smarr, C. E. Catlett // Communications of the ACM. — 1992. — Vol. 35, no. 6. — P. 44—52.

165. SNS-Based Secret Sharing Scheme for Security of Smart City Communication Systems [Текст] / A. Gladkov [et al.] // Ibero-American Congress of Smart Cities. — Springer. 2022. — P. 248—263.

166. *Soomro, Z. A.* Information security management needs more holistic approach: A literature review [Текст] / Z. A. Soomro, M. H. Shah, J. Ahmed // International journal of information management. — 2016. — Vol. 36, no. 2. — P. 215—225.

167. Specificity and affinity quantification of protein–protein interactions [Текст] / Z. Yan [et al.] // Bioinformatics. — 2013. — Vol. 29, no. 9. — P. 1127—1133.

168. *Spragins, J.* Learning without a teacher [Текст] / J. Spragins // IEEE Transactions on Information Theory. — 1966. — Vol. 12, no. 2. — P. 223—230. — URL: https://ieeexplore.ieee.org/abstract/document/1053865/?casa_token=gDfkVD9vTgoAAAAA:y4uYo75yC8KEctEV18cWK_W58lXwfaWrlZTiqqejwPmUTct-98_YNTgnCSj7Du8zDXU50vpSpVviwP4 (visited on 10/10/2023) ; Publisher: IEEE.

169. *Srinivas, M.* Genetic algorithms: A survey [Текст] / M. Srinivas, L. M. Patnaik // computer. — 1994. — Vol. 27, no. 6. — P. 17—26.

170. State-of-the-art in artificial neural network applications: A survey [Текст] / O. I. Abiodun [et al.] // Heliyon. — 2018. — Vol. 4, no. 11.

171. *Stojmenovic, I.* The fog computing paradigm: Scenarios and security issues [Текст] / I. Stojmenovic, S. Wen // 2014 federated conference on computer science and information systems. — IEEE. 2014. — P. 1—8.

172. *Strand, M.* A verifiable shuffle for the GSW cryptosystem [Текст] / M. Strand // Financial Cryptography and Data Security: FC 2018 International Workshops, BITCOIN, VOTING, and WTSC, Nieuwpoort, Curaçao, March 2, 2018, Revised Selected Papers 22. — Springer. 2019. — P. 165—180.

173. *Tang, J.* Artificial intelligence-based e-commerce platform based on SaaS and neural networks [Текст] / J. Tang // 2020 Fourth International Conference on Inventive Systems and Control (ICISC). — IEEE, 2020. — P. 421—424. — URL: https://ieeexplore.ieee.org/abstract/document/9171193/?casa_token=TmYwFdLDXq0AAAAA:8P5VVcZS_KWCXEnEm8xk2RPMV5kfWF27K9S9O9Z5fYh273EkseT7j0Jf7jZYAMOnPUX05sCbs (visited on 10/11/2023).

174. TenSEAL: A Library for Encrypted Tensor Operations Using Homomorphic Encryption [Текст] / A. Benaissa [et al.]. — 2021.

175. Tensorflow: Large-scale machine learning on heterogeneous distributed systems [Текст] / M. Abadi [et al.] // arXiv preprint arXiv:1603.04467. — 2016.

176. TFHE-rs: A library for safe and secure remote computing using fully homomorphic encryption and trusted execution environments [Текст] / L. Brenna [et al.] // Array. — 2022. — Vol. 13. — P. 100118.

177. TFHE: fast fully homomorphic encryption over the torus [Текст] / I. Chillotti [et al.] // Journal of Cryptology. — 2020. — Vol. 33, no. 1. — P. 34—91.

178. The fog computing service for healthcare [Текст] / Y. Shi [et al.] // 2015 2nd International symposium on future information and communication technologies for ubiquitous healthCare (Ubi-HealthTech). — IEEE. 2015. — P. 1—5.

179. Utilizing fully homomorphic encryption to implement secure medical computation in smart cities [Текст] / X. Sun [et al.] // Personal and Ubiquitous Computing. — 2017. — Vol. 21, no. 5. — P. 831—839. — Publisher: Springer.

180. *Wang, S.-C.* Artificial Neural Network [Текст] / S.-C. Wang // Interdisciplinary Computing in Java Programming. — Boston, MA : Springer US, 2003. — P. 81—100. — URL: http://link.springer.com/10.1007/978-1-4615-0377-4_5 (visited on 10/10/2023).

181. *Yathiraju, N.* Investigating the use of an Artificial Intelligence Model in an ERP Cloud-Based System [Текст] / N. Yathiraju // International Journal of Electrical, Electronics and Computers. — 2022. — Vol. 7, no. 2. — P. 1—26. — URL: https://aipublications.com/uploads/issue_files/IJEEC-APR20222-Investigating.pdf (visited on 10/11/2023).

182. *Yi, S.* A survey of fog computing: concepts, applications and issues [Текст] / S. Yi, C. Li, Q. Li // Proceedings of the 2015 workshop on mobile big data. — 2015. — P. 37—42.

183. *Yi, S.* Security and privacy issues of fog computing: A survey [Текст] / S. Yi, Z. Qin, Q. Li // Wireless Algorithms, Systems, and Applications: 10th International Conference, WASA 2015, Qufu, China, August 10-12, 2015, Proceedings 10. — Springer. 2015. — P. 685—695.

184. zama-ai/concrete-ml: Concrete ML: Privacy Preserving ML framework using Fully Homomorphic Encryption (FHE), built on top of Concrete, with bindings to traditional ML frameworks. [Текст]. — URL: https://github.com/zama-ai/concrete-ml (visited on 11/22/2024).

185. *Zhang, C.* Design and application of fog computing and Internet of Things service platform for smart city [Текст] / C. Zhang // Future Generation Computer Systems. — 2020. — Vol. 112. — P. 630—640.

186. *Zhang, P.* Security and trust issues in fog computing: A survey [Текст] / P. Zhang, M. Zhou, G. Fortino // Future Generation Computer Systems. — 2018. — Vol. 88. — P. 16—27.

Список рисунков

Список таблиц