УДК 004.891

АСПЕКТЫ РАЗРАБОТКИ АРХИТЕКТУРЫ ВОПРОСНО-ОТВЕТНОЙ СИСТЕМЫ ДЛЯ ОБРАБОТКИ БОЛЬШИХ ДАННЫХ НА ОСНОВЕ НЕЙРОСЕТЕВОГО МОДЕЛИРОВАНИЯ

Л. А. Демидова, д.т.н., профессор кафедры корпоративных информационных систем Института информационных технологий МИРЭА – Российского технологического университета, Москва, Россия; orcid.org/0000-0003-4516-3746, e-mail: demidova.liliya@gmail.com
Н. А. Морошкин, аспирант кафедры корпоративных информационных систем Института информационных технологий МИРЭА – Российского технологического университета, Москва, Россия; orcid.org/0009-0002-8787-2452, e-mail: moroshkin@mirea.ru

Рассмотрены аспекты нейросетевого вопросно-ответного моделирования на основе визуальных данных, приобретающего в последнее время всё большую востребованность. ***Целью работы*** *является разработка и обоснование архитектуры вопросно-ответной системы в контексте работы с end-to-end приложением или потоковым приложением для обработки больших данных. Представлены две архитектуры вопросно-ответной системы для end-to-end запросов и для потоковой обработки данных в контексте решаемой задачи нейросетевого вопросно-ответного моделирования на основе визуальных данных. Сформулированы рекомендации по выбору компонентов программного обеспечения, а именно: хранилищ исходных данных, сервисов развертывания и поддержки нейросетевых моделей, сервисов-брокеров очередей, столбчатых СУБД и in-memory СУБД для хранения уникальных данных в зависимости от специфики проектируемой вопросно-ответной системы*

Ключевые слова: *визуальное вопросно-ответное моделирование, мультимодальность, искусственные нейронные сети, end-to-end архитектура, архитектура потоковой обработки данных.*

DOI: 10.21667/1995-4565-2023-86-145-155

Введение

Вопросно-ответное моделирование (ВОМ, Question Answering, QA) давно и широко применяется при решении различных прикладных задач [1]. Однако, начиная с 2017 года [2], специалисты-разработчики начали проявлять интерес к синтезу вопросно-ответных систем (ВОС) с использованием технологий искусственного интеллекта, а именно: с применением нейросетевого моделирования. При этом особое внимание стало уделяться визуальному вопросно-ответному моделированию (ВВОМ, Visual Question Answering, VQA), которое обычно предполагает решение мультимодальной задачи обработки больших данных. В случае работы с моделями на основе алгоритмов машинного и глубокого обучения, в том числе и с нейросетевыми моделями, принято различать модальности, опираясь на природу данных (тип данных), используемых для формирования модальностей. В частности, модальности могут извлекаться из визуальных данных (изображений), текстовых данных (текстовых документов), аудиоданных и т.д.

В современной парадигме высокоуровневых исследований и нейросетевого моделирования понятие «большие данные» приобретает ключевое значение. Это понятие отражает масштабность данных различных типов, в частности аудиовизуальных, текстовых и числовых. «Большие данные» служат источником данных для обучения и настройки сложных нейронных архитектур в контексте решения задачи ВВОМ.

В случае ВВОМ целесообразно рассмотреть и исследовать в первую очередь модальности, формируемые на основе изображений и текстовых документов, представленных вопросами и ответами на них.

Постановка задачи ВВОМ в общем виде может быть сформулирована следующим образом. Пусть имеется множество единиц данных модальности на основе изображений $\{I_k\}$ $k=\overline{1,K}$, где K – число изображений в ВОС) и множества единиц данных модальности на основе текстовых документов-вопросов $\{Q_m\}$ $m=\overline{1,M}$, где M – число заданных к ВОС вопросов). Требуется сформировать множество единиц данных модальности на основе текстовых документов-ответов $\{A_m\}$ ($m=\overline{1,M}$ где M – число полученных от ВОС ответов), определив отображение VQA вида:

$$\{A_m\}=VQA\left(I_k,\{Q_m\}\right) \quad (1)$$

Следует отметить, что на одну единицу данных модальности на основе изображений может быть задано несколько вопросов, соответствующих единицам данных модальности на основе текстовых документов-вопросов, и на каждый из них ВОС должна дать осмысленный ответ, соответствующий единице данных модальности на основе текстовых документов-ответов. В данной работе не рассматривается случай, когда ВОС дает несколько ответов на один вопрос.

Вопросно-ответная система представляет собой систему, которая на основе анализа одной модальности способна ответить на вопрос, заданный в другой модальности (чаще всего в модальности, представленной текстовым документом). Работа с несколькими модальностями подразумевает построение архитектуры вопросно-ответной системы, выполнение специфичной настройки отдельных модулей и их конфигураций и выбор программного обеспечения.

Архитектура вопросно-ответной системы

В настоящее время предложен ряд вариантов решения задачи вопросно-ответного моделирования [3-5], в которых архитектура построена на основе базовой архитектуры, представленной в первой работе о нейросетевом вопросно-ответном моделировании [2]. Конфигурирование двух модальностей, сопоставленных экземплярам изображений и текстов, в базовой архитектуре может быть выполнено различным образом. В предлагаемой работе в контексте задачи вопросно-ответного моделирования в качестве данных для анализа, выполняемого в соответствии с некоторым вопросом с целью формирования ответа на него, будут выступать визуальные данные в виде некоторого экземпляра изображения, вопрос и ответ будут представлять собой экземпляры текста.

В базовой архитектуре [2] вопросно-ответной системы выделено три независимых модуля анализа: модуль визуального анализа, модуль анализа вопросов и QA-модуль (*Question-Answering* модуль). Модуль визуального анализа целесообразно реализовать на основе нейронной сети с применением архитектуры «трансформер» [4], используемой в качестве энкодера изображений. Модуль анализа вопросов также может быть реализован на основе нейронной сети с применением архитектуры «трансформер» [4], используемой в качестве энкодера текстовой информации. QA-модуль, генерирующий ответ на основе двух векторов признаков, соответствующих двум модальностям – на основе изображений и на основе текстовых документов-вопросов целесообразно реализовать на основе нейронной сети с применением многослойной нейронной сети в случае, если ответы – это вероятности наличия объектов из вопроса на изображении [3], или на основе нейронной сети с применением архитектуры «трансформер», используемой в качестве декодера в случае, если требуется получить текстовый ответ на вопрос [5-6].

Базовая архитектура ВОС [2] основана на классическом «мешке слов», используемом для анализа вопросов, и архитектуре рекуррентных нейронных сетей с долгой краткосрочной памятью, а именно архитектуре LSTM (Long Short-Term Memory), применяемой для анализа изображений. Базовая архитектура ВОС может быть использована для ВВОМ (1) на основе искусственных нейронных сетей.

Зависимость VQA множества ответов $\{A_m\}$ ($m=\overline{1,M}$, где M – число полученных от ВОС ответов) от множества вопросов $\{Q_m\}$ ($m=\overline{1,M}$, где M – число заданных к ВОС вопросов) и исходного изображения I_k ($k=\overline{1,K}$, где K – число изображений в ВОС) может быть представлена в виде функции получения ответа QA из множества векторов признаков (вложений в векторное пространство, embeddings) вопросов $\{Q_m^{emb}\}$ и вектора признаков изображения I_m^{emb}:

$$VQA(Q_m, I_k) = QA\left(I_k^{emb}, \{Q_m^{emb}\}\right) \tag{2}$$

Пусть векторы изображений можно получить с помощью функции преобразования V, а векторы вопросов можно получить функцией преобразования T. Тогда итоговая постановка задачи ВВОМ имеет вид:

$$VQA(Q_m,\ I_k) = QA\left(I_k^{emb}, \{Q_m^{emb}\}\right) = QA\left(T\left(\{Q_m\}\right),\ V\left(I_k\right)\right) \tag{3}$$

Исходя из постановки задачи (3), можно сделать следующие выводы.

1. Работа вопросно-ответной системы стандартной архитектуры является синхронной, то есть QA-модуль обрабатывает совместно вектор признаков изображения и множество векторов признаков вопросов.

2. Время работы вопросно-ответной системы зависит от времени работы её самого медленного компонента, отвечающего за преобразование входных данных в векторы признаков модуля анализа вопросов или модуля визуального анализа.

Выбор программного обеспечения в зависимости от типа приложения

Входные данные для ВОС могут быть получены как от end-to-end приложения, так и от потокового приложения.

В случае работы с end-to-end приложением входные данные будут появляться в ВОС по запросу от внешних систем, т.е. будет наблюдаться дискретное поступление в ВОС. В случае работы с потоковым приложением входные данные для ВОС будут представлять собой непрерывный поток данных.

Обработка end-to-end запросов

End-to-end приложение в контексте ВОС представляет собой систему, которая соблюдает приоритет входных сообщений и отдает ответ только по запросу. Пропускная способность любого модуля рассматриваемой ВОС – характеристика, показывающая число проходящих через ВОС пар единиц изображений и множество вопросов к этой единице изображения. Время работы любого модуля ВОС – это время, за которое модуль способен обработать всю входную информацию и выдать ответ.

Исходя из постановки задачи ВВОМ (3), для решения которой будет выполняться разработка архитектуры ВОС, целесообразно определить метрику (качественную характеристику), которая позволит оценить архитектуру при её реализации через время работы t_{all} всей системы. Это время прямо пропорционально произведению коэффициента пропорциональности k_t и суммы времени работы t_{QA} QA-модуля и максимального значения из времени работы модуля визуального анализа t_V и времени работы модуля анализа вопросов t_Q:

$$t_{all} = k_t * (t_{QA} + \max\left(t_V, t_Q\right)) \tag{4}$$

Архитектура ВОС должна обеспечивать минимально возможную задержку при выдаче ответов на вопросы (запросы к ВОС). Возможная задержка ответа на запрос обусловлена множеством факторов [7], в том числе характеристиками выбранного программного обеспечения.

В качестве программной архитектуры для базовой задачи ВВОМ (3) целесообразно использовать архитектуру программного приложения REST [8], а в качестве основного протокола для передачи информации выбрать HTTP протокол, являющийся базовым для архитектуры REST [8]. Выбор архитектуры приложения REST может быть обоснован целесообразностью использования её основных принципов [9]: приложение REST основано на клиент-серверной архитектуре, поэтому новое приложение, архитектура которого подобна архитектуре приложения REST, будет поддерживать масштабируемость и гибкость в построении логики обработки данных.

Обычно стандартная схема взаимодействия клиента (как поставщика данных в ВОС) и сервера (ВОС) предполагает, что клиент сначала отправляет один GET-запрос, где указывает исходное изображение, список всех вопросов и метаданные, необходимые для идентификации исходного изображения и вопросов, а затем в качестве ответа на GET-запрос получает от ВОС текстовый ответ. Однако в случае задачи ВВОМ использовать такую схему взаимодействия нецелесообразно [6], так как время обработки изображения и вопросов может превышать принятые стандарты ответа на GET-запрос [9]. В связи с этим в рассматриваемой ВОС целесообразно использовать схему запросов, в которой клиент отправляет к ВОС POST-запрос, содержащий исходные данные (вопросы и изображение), а в ответ на POST-запрос сервер (ВОС) отправляет клиенту метаданные, необходимые для идентификации ответа. После этого клиент способен получить текстовый ответ на исходное изображение и вопросы с помощью GET-запроса.

Так как передача изображений в исходном формате требует большого объема памяти от протокола передачи, целесообразно передавать изображение в байтовом виде, что позволяет уменьшить объем памяти, необходимый для передачи данных. Для реализации работы с байтовыми изображениями в рамках проектируемой архитектуры ВОС целесообразно использовать In-Memory базы данных, например Redis [10] или Dragonfly [11].

Redis представляет собой NoSQL базу данных, использующую структуру «ключ-значение» для реализации хранения данных. Redis максимально оптимальна для атомарных операций, что подходит для чтения и записи байтовых изображений.

Dragonfly [9] является системой кэширования и хранения данных в оперативной памяти, обеспечивающей интенсивное использование данных и низкую задержку на запросы. Сравнительный анализ Redis и Dragonfly по времени работы (в смысле быстродействия) показывает преимущество Dragonfly по отношению к Redis более, чем в 30 раз [11].

В модуле анализа вопросов в качестве основной модальности данных выступают неструктурированные или слабоструктурированные тексты (текстовые документы). В связи с этим для оптимального хранения данных и для сокращения необходимого объема памяти по передаче информации целесообразно использовать семейство колоночных баз данных. Если сравнить стандартный подход РСУБД с хранением данных в строках с колоночным семейством NoSQL в контексте решения задачи ВВОМ, то предпочтение следует отдать колоночному семейству NoSQL [12], так как СУБД, где данные хранятся в формате столбцов, наиболее оптимальны для хранения больших данных и множественных SELECT-запросов.

В качестве программного обеспечения для реализации хранения и обработки данных в проектируемой архитектуре ВОС следует рассмотреть программные продукты Apache Cassandra [13], являющейся частью экосистемы Hadoop [13], и столбчатую СУБД Clickhouse [14].

Apache Hadoop представляет собой экосистему и фреймворк для работы с большими данными, из экосистемы Hadoop для хранения данных целесообразно использовать колоночную СУБД Apache Cassandra [13], которая представляет собой распределенную систему хранения данных для управления большими объемами структурированных данных.

Кроме того, для реализации хранения и обработки данных в проектируемой архитектуре ВОС можно использовать столбцово-ориентированную СУБД Clickhouse [14]. В этой СУБД данные при хранении группируются по столбцам, а при выполнении запросов – в массивах (векторах или фрагментах столбцов). В контексте разрабатываемой ВОС Clickhouse может

быть использована не только для хранения вопросов, но и для хранения ответов ВОС. У СУБД Clickhouse имеется несколько преимуществ перед СУБД Apache Cassandra: СУБД Clickhouse предоставляет интерфейс для подключения [14] таблиц РСУБД (например, MariaDB или PostgreSQL) и возможность интеграции в триггер-функции программных скриптов [14], реализованных на разных языках программирования (Python, Go, Java, JavaScript).

Одна из самых частых проблем, с которой сталкиваются разработчики при построении программной архитектуры end-to-end приложения (рисунок 1) для обработки больших данных, является дубликация входных данных. В связи с этим особое внимание при разработке ВОС следует уделить обеспечению функционала удаления дублей входных визуальных данных. Для решения проблемы дубликации применяются различные алгоритмы дедубликации или удаления дубликатов [15]. Базовые алгоритмы, используемые для решения проблемы дубликации, основаны на подсчете хеш-суммы [16] в байтовом представлении исходных файлов.

При работе с текстовыми данными дедубликацию можно реализовать с использованием внутренних инструментов СУБД. В рассматриваемой ВОС целесообразно использовать встроенные в СУБД Clickhouse или СУБД Apache Cassandra стратегии дедубликации [16].

При работе с изображениями также следует учитывать возможность дубликации входных данных и как следствие дублирование работы всей ВОС. Известно большое число стратегий дедубликации [16], но практически все они основаны на работе с байтовым представлением исходных данных. In-Memory базы данных Dragonfly и Redis, рассмотренные выше, позволяют удалять дубли в записанных данных.

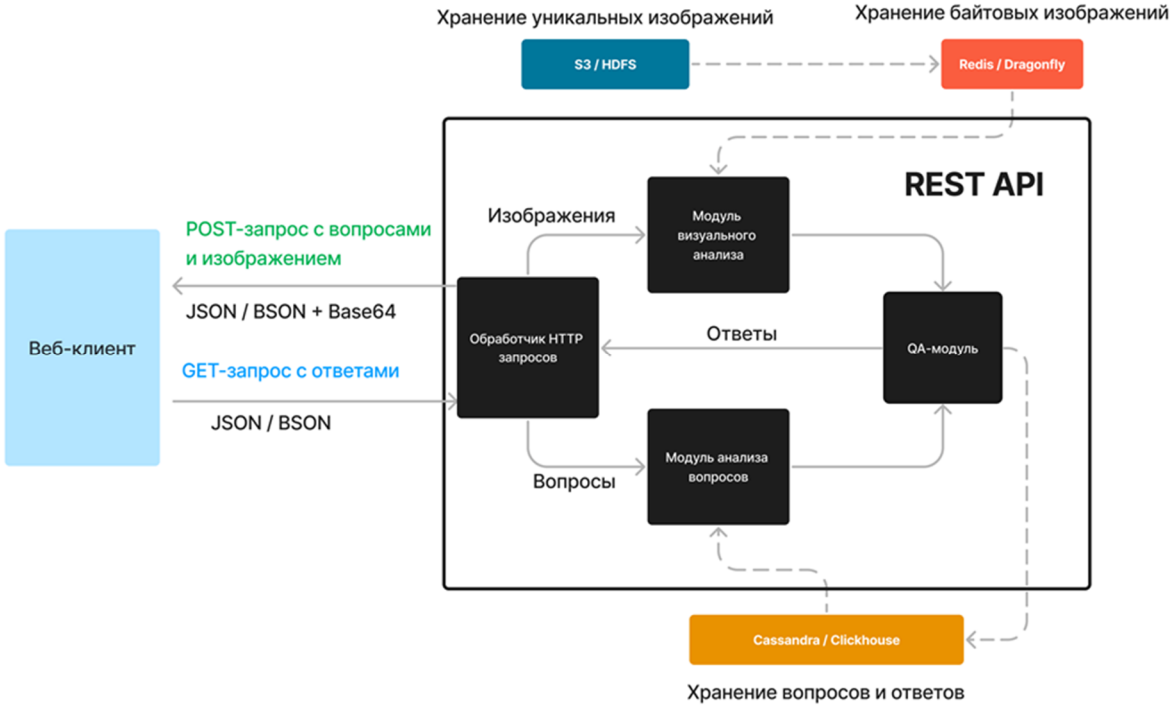

Рисунок 1 – Предлагаемая архитектура вопросно-ответной системы с использованием архитектуры REST
Picture 1 –Proposed architecture of a question-answering system using REST architecture

Так как Redis и Dragonfly являются in-memory хранилищами, то работа с данными внутри них ограничена параметром «время жизни» данных (*Time To Live, TTL*). Для реализации постоянного хранения визуальных данных предлагаемой архитектуры ВОС (рисунок 1) целесообразно использовать распределенные хранилища данных [17], в частности, может быть использовано хранилище Amazon S3 (Amazon Simple Storage Service), представляющее собой облачное объектное хранилище данных, предназначенное для хранения большого объема данных [18]. Последние работы по Amazon S3 [18–20] доказывают его приспособленность

для решения задачи, стоящей перед ВОС. Amazon S3 поддерживает большинство методик шардирования данных, что важно в разрезе больших данных. В качестве аналога Amazon S3 следует назвать систему Apache HDFS [21], являющуюся частью экосистемы Apache Hadoop. Apache HDFS представляет собой распределенную файловую систему, созданную для хранения больших данных [21]. Она поддерживает большинство форматов для хранения больших файлов (Parguet, AVRO, ORC и другие) [21]. Достоинствами Apache HDFS являются отказоустойчивость (благодаря реализации реплицирования данных [21] внутри файлового хранилища HDFS) и возможность горизонтального и вертикального масштабирования [21].

При осуществлении выбора между Amazon S3 и Apache HDFS с целью хранения данных необходимо ориентироваться на специфику конкретной ВОС [22]. В случае, если главными требованиями к ВОС являются консистентность данных и реплицирование, целесообразно использовать Apache HDFS [20]. Если же проектируемая ВОС требует масштабируемости и надежности, то следует использовать Amazon S3 [22].

Обработка потоковых данных

Последние работы в разрезе вопросно-ответного моделирования на основе нейросетей [6, 20, 21] показывают универсальность вопросно-ответных архитектур для большинства мультимодальных задач, так как ВОС способна решать не только задачу ВВОМ, но и задачу текстового описания визуальных данных, а также задачу классификации визуальных и текстовых данных. При решении задач описания и классификации в системе ВОС чаще всего выполняют потоковую обработку данных [22], что говорит о важности рассмотрения аспектов построения архитектуры ВОС в контексте потоковой обработки (рисунок 2).

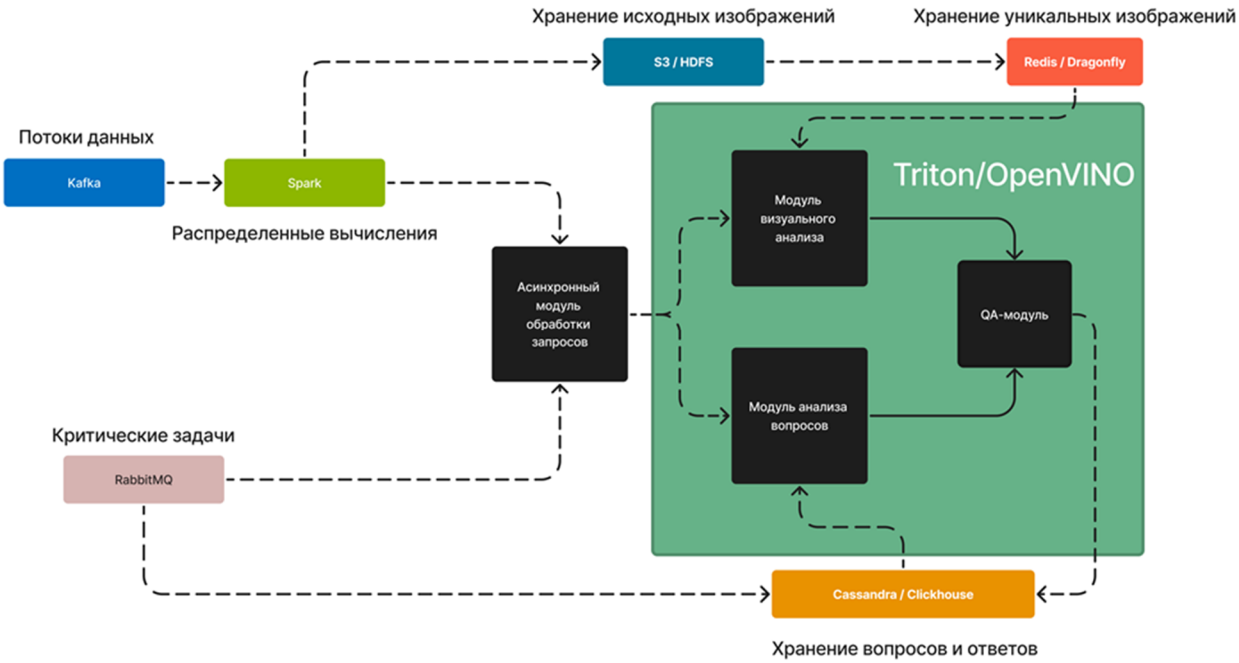

Рисунок 2 – Предлагаемая архитектура вопросно-ответной системы для обработки потоковых данных

Picture 2 – Proposed architecture of a question-answering system for processing streaming data

Потоковая обработка данных предполагает работу с непрерывным и постоянным потоком задач к ВОС. Для реализации потока данных целесообразно использовать брокеры очередей, например Apache Kafka [23] и RabbitMQ [24] на основе протокола MQTT и стандарта AMQP.

Apache Kafka представляет собой распределенную платформу потоковой передачи событий с открытым исходным кодом, предназначенную для потоковой обработки данных. Тип архитектуры Apache Kafka – «Smart Client / Dumb Server» [25]. Архитектура «Smart Client / Dumb Server» предполагает взаимодействие программного клиента с сервером таким обра-

зом, что клиент автоматически подбирает необходимую нагрузку на выделенные ресурсы [26] в зависимости от общего числа запросов к клиенту.

RabbitMQ является брокером сообщений, основанным на протоколе MQTT и стандарте AMQP. Он построен на архитектуре «Dumb Client / Smart Server». RabbitMQ используется для брокинга сообщений-событий с выделением приоритета, он обладает возможностью создания гибкой логики взаимодействия клиента-слушателя и сервера-брокера [25].

Так как Apache Kafka и RabbitMQ реализуют разное взаимодействие с программными клиентами, учитывая этот факт [24, 25], целесообразно использовать Apache Kafka для реализации потока данных для ВОС, а RabbitMQ для хранения и подготовки критических сообщений или запросов к ВОС.

Брокеры сообщений реализуют лишь хранение, подготовку и интерфейс данных для обработки данных вопросно-ответной системой. Для реализации распределенной обработки данных необходимо использовать дополнительное программное обеспечение. Для потоковой обработки неструктурированных и слабоструктурированных данных (в частности, текстовых вопросов в контексте ВОС) в проектируемой архитектуре (рисунок 2) предлагается использовать приложения на основе библиотеки Apache Spark [27], представляющей собой фреймворк с открытым исходным кодом для реализации распределённой обработки неструктурированных и слабоструктурированных данных. В контексте потоковых данных Apache Spark позволяет увеличить скорость обработки данных по сравнению с аналогами.

При потоковой обработке данных в контексте нейросетевого ВВОМ необходимо реализовать оптимизацию обработки данных каждым модулем ВОС (рисунок 2).

Так как в основе каждого модуля лежит нейронная сеть, целесообразно использовать стратегии оптимизации работы нейросетей. Известно множество подходов к оптимизации нейронных сетей, среди которых необходимо в первую очередь назвать такие, как квантизация весов модели [27] и оптимизация архитектуры нейронной сети [28]. Однако наибольший прирост в скорости обработки данных (в увеличении пропускной способности ВОС) предоставляют сервисы поддержки и развертывания нейронных сетей, в частности OpenVINO [28] и Nvidia Triton [29].

OpenVINO представляет собой сервис для поддержки и развертывания нейронных сетей для центральных процессоров Intel, позволяющий развертывать, оптимизировать и поддерживать нейронные сети. OpenVINO обычно используется для оптимизации работы нейронных сетей, работающих с текстовыми данными [28].

При работе с мультимодальными данными, если используется модальность на основе изображений, целесообразно использовать GPU-ускорители. Одним из самых популярных решений в ускорении работы моделей на GPU и интеллектуального развертывания нейронных сетей является Nvidia Triton [29]. NVIDIA Triton представляет собой оптимизированное для вывода программное обеспечение с открытым кодом, которое поддерживает различные платформы машинного обучения, в частности TensorFlow, ONNX Runtime, PyTorch, Nvidia TensorRT [30].

Nvidia Triton представляет собой аналогичный OpenVINO сервис, ориентированный на работу с GPU-ускорителеми и способный ускорить работу нейросетевых моделей и предоставить интерфейс для взаимодействия с ними. Нейронные сети, развернутые в Nvidia Trtion работают как микросервисы [28], сам же сервер предоставляет удобный доступ к ним. Для обращения к Nvidia Triton можно использовать HTTP и gRPC протоколы. В случае большого потока данных целесообразно использовать gRPC протокол, так как он более приспособлен к потоковой обработке данных [28] и позволяет обрабатывать данные распределенно.

Архитектура ВОС потребует дополнительной логики в случае использования Nvidia Triton, так как модули ВОС в Nvidia Triton представляют собой микросервисы, между которыми взаимодействие осуществляется через технологию shared memory [31]. В этом случае целесообразно реализовать дополнительный модуль логики (рисунок 2), который будет осуществлять связь между модулями ВОС.

Главное различие между end-to-end архитектурой и потоковой архитектурой ВОС заключается в способности ВОС обеспечивать максимальную пропускную способность в зависимости от входных данных. В случае архитектуры для end-to-end запросов необходимо учитывать непостоянство входных данных (заключающееся в том, что обращение к системе создается клиентами только по запросу конечных пользователей) и обеспечивать максимально быстрое реагирование всей ВОС на входной запрос. В случае архитектуры для потоковой обработки данных главная задача программного обеспечения заключается в обработке входного потока данных с минимальной задержкой обработки при недопущении накопления сообщений с исходными данными в потоке.

Заключение

В данной работе была поставлена задача ВВОМ (3), были предложены две архитектуры ВОС для решения задачи ВВОМ для end-to-end приложений (рисунок 1) и для потоковой обработки данных (рисунок 2) в случае использования нейросетевого моделирования. Описаны и представлены главные преимущества выбранного программного обеспечения, предложены альтернативы для возможности гибкого выбора программных продуктов с учетом специфики задачи ВВОМ. Сформулированы рекомендации по выбору хранилищ текстовых данных (Apachhe Cassandra и Clickhouse), по выбору сервисов развертывания и поддержки нейросетевых моделей при обработке потоковых данных (OpenvVINO и Nvidia Triton), по выбору сервисов-брокеров сообщений для потоковых данных (RabbitMQ и Apache Kafka), по выбору хранилищ исходных визуальных данных (Apache HDFS и AWS S3) и по выбору in-memory СУБД для хранения уникальных визуальных данных (Redis и Dragonfly).

Цель дальнейших исследований – создание нейросетевой архитектуры модулей ВОС, обзор современных архитектур нейронных сетей для вопросно-ответного моделирования, подготовка и обучение нейронных моделей, способных решать задачу ВВОМ (3).

Библиографический список

1. **Hung-Ting Chen, Fangyuan Xu, Shane A. Arora, Eunsol Choi** Understanding Retrieval Augmentation for Long-Form Question Answering // Under review as a conference paper at ICLR 2024, 2023. pp. 1-3.

2. **Aishwarya Agrawal, Jiasen Lu, Stanislaw Antol, Margaret Mitchell, C. Lawrence Zitnick, Dhruv Batra, Devi Parikh** VQA: Visual Question Answering // International Conference on Computer Vision (ICCV) 2015, 2015. pp. 1-13.

3. **Vahid Kazemi Ali Elqursh** Show, Ask, Attend, and Answer: A Strong Baseline for Visual Question Answering, 2023. pp. 1-7.

4. **Zhengyuan Yang, Linjie Li, Kevin Lin, Jianfeng Wang, Chung-Ching Lin, Zicheng Liu, Lijuan Wang** The Dawn of LMMs: Preliminary Explorations with GPT-4V(ision), 2023. pp. 12-63.

5. **Jiwan Chung, Youngjae Yu** VLIS: Unimodal Language Models Guide Multimodal Language Generation, 2023. pp. 1-22.

6. **Junnan Li, Dongxu Li, Caiming Xiong, Steven Hoi** BLIP: Bootstrapping Language-Image Pre-training for Unified Vision-Language Understanding and Generation, 2022. pp. 1-12.

7. **Mudasir Ahmad Wani, Suraiya Jabin** Big Data: Issues, Challenges, and Techniques in Business Intelligence // In book: Big Data Analytics (pp.613-628), 2018.

8. **Yifan Song, Weimin Xiong, Dawei Zhu, Wenhao Wu, Han Qian, Mingbo Song, Hailiang Huang, Cheng Li, Ke Wang, Rong Yao, Ye Tian, Sujian Li** RestGPT: Connecting Large Language Models with Real-World RESTful APIs, 2023. pp. 5-25.

9. **Yasmin Makki, Nadia Mahmood Hussien, Hanan Abed** RESTful API Design for a Real-Time Weather Reporting System // In book: Intelligent Systems and Networks (pp.557-565)

10. **Dirk Eddelbuettel** A Brief Introduction to Redis, 2022. pp. 1-3.

11. **Gulayeva N.M., Mykhailo Kobieliev** Implementation of FSM Based Chat-Bots in a Graphical Designer // NaUKMA Research Papers Computer Science, 2023. pp. 31-40.

12. **Fouad Elotmani, Redouane Esbai, Mohamed Atounti** Creation of column-oriented NoSQL databases automatically in Big Data environments and its impact on energy consumption // E3S Web of Conferences, 2023. pp. 412-421.

13. **Avinash Lakshman Prashant Malik Cassandra** A Decentralized Structured Storage System // ACM SIGOPS Operating Systems Review, 2010. pp. 35-40.

14. **Fazl Barez, Paul Bilokon, Ruijie Xiong** Benchmarking Specialized Databases for High-frequency Data, 2023. pp. 1-29.

15. **Баранчиков А.И., Яковлев И.И., Клюева И.А.** Использование методов очистки данных при реинжиниринге баз данных // Вестник Рязанского государственного радиотехнического университета. 2021. №76. С. 35-42.

16. **Nidhi B Sankhe, Shweta Singh, Sumedh Pundkar** Removal of Duplicate Data using Secure Hash and Difference Hash Algorithm // International Research Journal of Engineering and Technology (IRJET), 2021. pp. 13-17.

17. **Ушаков Ю.А.** Исследование производительности распределенной виртуальной программно-конфигурируемой инфраструктуры для задач обработки больших данных // Вестник Рязанского государственного радиотехнического университета. 2021. № 77. С. 58-68.

18. **James Bornholt, Rajeev Joshi, Vytautas Astrauskas, Brendan Cully, Bernhard Kragl** Using Lightweight Formal Methods to Validate a Key-Value Storage Node in Amazon S3 // Proceedings of the ACM SIGOPS 28th Symposium on Operating Systems Principles, 2021. pp. 836-850.

19. **Balakrishna Phani Kommanaboina, Aditya Goverdhana, Naveena Kanderi, Jagadeesh Karri** A Technical Report on Image Classification using AWS, 2021. pp. 1-19.

20. **Raphael Shu, Elman Mansimov, Tamer Alkhouli, Nikolaos Pappas, Salvatore Romeo, Arshit Gupta, Saab Mansour, Yi Zhang, Dan Roth** Dialog2API: Task-Oriented Dialogue with API Description and Example Programs, 2022. pp. 1-15.

21. **Liwei Tian, Yu Sun, Lei Yang** Overview of Storage Architecture and Strategy of HDFS // In book: Applied Mathematics, Modeling and Computer Simulation, 2022.

22. **Alaa Jamal, Rita Fleiner, Eszter Kail** Performance Comparison between S3, HDFS and RDS storage technologies for real-time big-data applications // 2021 IEEE 15th International Symposium on Applied Computational Intelligence and Informatics (SACI), 2021.

23. **Shashank Kumar, Sachin Sharma, Aryan Jadon** Distributed Kafka Clusters: A Novel Approach to Global Message Ordering, 2023. pp. 1-6.

24. **Gary Weiss, Jeffrey Lockhart** A comparison of alternative client/server architectures for ubiquitous mobile sensor-based applications // Proceedings of the 2012 ACM Conference on Ubiquitous Computing, 2012. pp. 721-724.

25. **Jonathan Will, Lauritz Thamsen, Dominik Scheinert, Odej Kao** Selecting Efficient Cluster Resources for Data Analytics: When and How to Allocate for In-Memory Processing? // ACM SSDBM 2023, 2023. pp. 1-4.

26. **Ушакова М.В., Ушаков Ю.А.** Исследование сети виртуальной инфраструктуры центра обработки данных с гибридной программно-конфигурируемой коммутацией // Вестник Рязанского государственного радиотехнического университета. 2021. № 75. С. 34-44.

27. **Saleh Ashkboos, Ilia Markov, Elias Frantar, Tingxuan Zhong, Xincheng Wang, Jie Ren, Torsten Hoefler, Dan Alistarh** Towards End-to-end 4-Bit Inference on Generative Large Language Models, 2023. pp. 1-9

28. **Pedro Machado** Computational models of object motion detectors accelerated using FPGA technology // PhD Thesis, 2023, pp. 10-42.

29. **Philippe Tillet, H. T. Kung, David Cox** Triton: An Intermediate Language and Compiler for Tiled Neural Network Computations // Proceedings of the 3rd ACM SIGPLAN International Workshop on Machine Learning and Programming Languages, 2019. pp. 10-19.

30. **Hoang Viet Pham, Thinh Gia Tran, Chuong Dinh Le, An Dinh Le, Hien Bich Vo** Benchmarking Jetson Edge Devices with an End-to-end Video-based Anomaly Detection System // Accepted in Future of Information and Communication Conference (FICC) 2024, 2023. pp. 1-17.

31. **Yong Liu, Hang Dong, Boyang Liang, Songwei Liu, Qingi Dong Unfolding** Once is Enough: A Deployment-Friendly Transformer Unit for Super-Resolution // ACM International Conference on Multimedia, 2023. pp. 1-15.

UDC 004.891

ASPECTS OF THE DEVELOPMENT OF QUESTION AND ANSWER SYSTEM ARCHITECTUREFOR BIG DATA PROCESSING BASED ON NEURAL NETWORK MODELING

L. A. Demidova, Dr. in technical sciences, Full Professor, Department of Corporate Information Systems, Institute of Information Technologies, MIREA – Russian Technological University, Moscow, Russia;
orcid.org/0000-0003-4516-3746, e-mail: demidova.liliya@gmail.com
N. A. Moroshkin, post-graduate student, Department of Corporate Information Systems, Institute of Information Technologies, MIREA – Russian Technological University, Moscow, Russia;
orcid.org/0009-0002-8787-2452, e-mail: whiteandblackreality@gmail.com

The article discusses the aspects of neural network question-answer modeling based on visual data, which has recently become increasingly popular. ***The aim of this work*** *is to develop and justify the architecture of a question-answering system in the context of working with streaming big data or end-to-end applications. Two architectures of a question-answer system are presented for end-to-end queries and for stream processing in the context of the problem being solved of neural network question-answer modeling based on visual data.*

Keywords: *visual question-answer modeling, multimodality, artificial neural networks, end-to-end architecture, stream data processing architecture.*

DOI: 10.21667/1995-4565-2023-86-145-155

References

1. **Hung-Ting Chen, Fangyuan Xu, Shane A. Arora, Eunsol Choi** Understanding Retrieval Augmentation for Long-Form Question Answering . *Under review as a conference paper at ICLR 2024*, 2023. pp. 1-3.

2. **Aishwarya Agrawal, Jiasen Lu, Stanislaw Antol, Margaret Mitchell, C. Lawrence Zitnick, Dhruv Batra, Devi Parikh** VQA: Visual Question Answering . *International Conference on Computer Vision (ICCV)* 2015, 2015, pp. 1-13.

3. **Vahid Kazemi, Ali Elqursh.** *Show, Ask, Attend, and Answer: A Strong Baseline for Visual Question Answering*, 2023, pp. 1-7.

4. **Zhengyuan Yang, Linjie Li, Kevin Lin, Jianfeng Wang, Chung-Ching Lin, Zicheng Liu, Lijuan Wang.** *The Dawn of LMMs: Preliminary Explorations with GPT-4V(ision)*, 2023. pp. 12-63.

5. **Jiwan Chung, Youngjae Yu** *VLIS: Unimodal Language Models Guide Multimodal Language Generation*, 2023, pp. 1-22.

6. **Junnan Li, Dongxu Li, Caiming Xiong, Steven Hoi** *BLIP: Bootstrapping Language-Image Pre-training for Unified Vision-Language Understanding and Generation*, 2022, pp. 1-12.

7. **Mudasir Ahmad Wani, Suraiya Jabin** Big Data: Issues, Challenges, and Techniques in Business Intelligence. In book: *Big Data Analytics* . 2018, pp.613-628, 2018.

8. **Yifan Song, Weimin Xiong, Dawei Zhu, Wenhao Wu, Han Qian, Mingbo Song, Hailiang Huang, Cheng Li, Ke Wang, Rong Yao, Ye Tian, Sujian Li.** *RestGPT: Connecting Large Language Models with Real-World RESTful APIs*, 2023, pp. 5-25.

9. **Yasmin Makki, Nadia Mahmood Hussien, Hanan Abed** RESTful API Design for a Real-Time Weather Reporting System. In book: *Intelligent Systems and Networks* (pp.557-565)

10. **Dirk Eddelbuettel.** *A Brief Introduction to Redis*, 2022, pp. 1-3.

11. **Gulayeva N.M., Mykhailo Kobieliev** Implementation of FSM Based Chat-Bots in a Graphical Designer. *NaUKMA Research Papers Computer Science*. 2023, pp. 31-40.

12. **Fouad Elotmani, Redouane Esbai, Mohamed Atounti** Creation of column-oriented NoSQL databases automatically in Big Data environments and its impact on energy consumption. *E3S Web of Conferences*, 2023, pp. 412-421.

13. **Avinash Lakshman Prashant Malik Cassandra** A Decentralized Structured Storage System. *ACM SIGOPS Operating Systems Review*, 2010, pp. 35-40.

14. **Fazl Barez, Paul Bilokon, Ruijie Xiong.** *Benchmarking Specialized Databases for High-frequency Data*, 2023. pp. 1-29.

15. **Baranchikov A.I., Yakovlev I.I., Klyueva I.A.** Using data cleaning methods in database reengineering. *Vestnik Ryazanskogo gosudarstvennogo radiotekhnicheskogo universiteta*, 2021, no. 76. pp. 35-42.

16. **Nidhi B Sankhe, Shweta Singh, Sumedh Pundkar** Removal of Duplicate Data using Secure Hash and Difference Hash Algorithm. *International Research Journal of Engineering and Technology* (IRJET), 2021. pp. 13-17.

17. **Ushakov Y.A.** Study of the performance of distributed virtual software-defined infrastructure for big data processing tasks. *Vestnik Ryazanskogo gosudarstvennogo radiotekhnicheskogo universiteta*. 2021, no. 77, pp. 58-68 (in Russian).

18. **James Bornholt, Rajeev Joshi, Vytautas Astrauskas, Brendan Cully, Bernhard Kragl** Using Lightweight Formal Methods to Validate a Key-Value Storage Node in Amazon S3. *Proceedings of the ACM SIGOPS 28th Symposium on Operating Systems Principles*, 2021, pp. 836-850.

19. **Balakrishna Phani Kommanaboina, Aditya Goverdhana, Naveena Kanderi, Jagadeesh Karri.** *A Technical Report on Image Classification using AWS*, 2021. pp. 1-19.

20. **Raphael Shu, Elman Mansimov, Tamer Alkhouli, Nikolaos Pappas, Salvatore Romeo, Arshit Gupta, Saab Mansour, Yi Zhang, Dan Roth.** *Dialog2API: Task-Oriented Dialogue with API Description and Example Programs*, 2022, pp. 1-15.

21. **Liwei Tian, Yu Sun, Lei Yang** Overview of Storage Architecture and Strategy of HDFS. *In book: Applied Mathematics, Modeling and Computer Simulation*, 2022.

22. **Alaa Jamal, Rita Fleiner, Eszter Kail** Performance Comparison between S3, HDFS and RDS storage technologies for real-time big-data applications. *2021 IEEE 15th International Symposium on Applied Computational Intelligence and Informatics (SACI)*, 2021.

23. **Shashank Kumar, Sachin Sharma, Aryan Jadon.** *Distributed Kafka Clusters: A Novel Approach to Global Message Ordering*, 2023, pp. 1-6.

24. **Gary Weiss, Jeffrey Lockhart** A comparison of alternative client/server architectures for ubiquitous mobile sensor-based applications. *Proceedings of the 2012 ACM Conference on Ubiquitous Computing*. 2012, pp. 721-724.

25. **Jonathan Will, Lauritz Thamsen, Dominik Scheinert, Odej Kao** Selecting Efficient Cluster Resources for Data Analytics: When and How to Allocate for In-Memory Processing? *ACM SSDBM 2023*. 2023, pp. 1-4.

26. **Ushakova M.V., Ushakov Y.A.** Research of the virtual infrastructure network of a data center with hybrid software-defined switching. *Vestnik Ryazanskogo gosudarstvennogo radiotekhnicheskogo universiteta*. 2021, no. 75, pp. 34-44. (in Russian).

27. **Saleh Ashkboos, Ilia Markov, Elias Frantar, Tingxuan Zhong, Xincheng Wang, Jie Ren, Torsten Hoefler, Dan Alistarh** Towards End-to-end 4-Bit Inference on Generative Large Language Models, 2023. pp. 1-9.

28. **Pedro Machado** Computational models of object motion detectors accelerated using FPGA technology. *PhD Thesis*. 2023, pp. 10-42.

29. **Philippe Tillet, H. T. Kung, David Cox** Triton: An Intermediate Language and Compiler for Tiled Neural Network Computations. *Proceedings of the 3rd ACM SIGPLAN International Workshop on Machine Learning and Programming Languages*. 2019, pp. 10-19.

30. **Hoang Viet Pham, Thinh Gia Tran, Chuong Dinh Le, An Dinh Le, Hien Bich Vo** Benchmarking Jetson Edge Devices with an End-to-end Video-based Anomaly Detection System. *Accepted in Future of Information and Communication Conference (FICC) 2024*. 2023, pp. 1-17.

31. **Yong Liu, Hang Dong, Boyang Liang, Songwei Liu, Qingi Dong Unfolding** Once is Enough: A Deployment-Friendly Transformer Unit for Super-Resolution. *ACM International Conference on Multimedia*. 2023, pp. 1-15.