УДК 004.75

АНАЛИЗ ПРИНЦИПОВ И МЕТОДОВ ФОРМИРОВАНИЯ ХРАНИЛИЩА ВРЕМЕННЫХ РЯДОВ И ОРГАНИЗАЦИИ ПРОГРАММНЫХ ИНТЕРФЕЙСОВ В РЕПОЗИТОРИИ НЕЙРОСЕТЕВЫХ МОДЕЛЕЙ

Ямашкин С.А., Скворцов М.А., Большакова М.В., Ямашкин А.А.

ФГБОУ ВО «Национальный исследовательский Мордовский государственный университет им. Н.П. Огарёва», Саранск, e-mail: dep-general@adm.mrsu.ru

В Российской Федерации остро стоит вопрос усиления связанности территории. Немаловажную роль в решении данного вопроса играют инфраструктуры пространственных данных, внедряемые для обработки и анализа информации о сложных геосистемных моделях. Ядром систем данного класса в настоящее время становятся алгоритмы машинного обучения, консолидируемые в репозиториях нейросетевых моделей и позволяющие автоматизированно выделять пространственно-временные закономерности и на базе имеющихся данных прогнозировать наступление или исход природных катаклизмов. Цели данного исследования – подробное изучение принципов работы InfluxDB, оценивание ее возможностей и рассмотрение возможности ее применения в качестве хранилища для репозитория нейросетевых моделей. InfluxDB работает при использовании технологии кеширования для сохранения мгновенного результата и файла восстановления и запуска системы WAL, который хранит основную структуру организации базы данных. Читается данный файл только в одном случае – с целью восстановления системы. Для того чтобы размер этого файла был меньше, к нему применяют библиотеку сжатия Snappy. Для хранения на диске используется система столбчатого хранения данных TSM. Данный формат основан на построении логарифмически структурированных деревьев слияний. При использовании данной структуры файлы обрабатывают по мере сброса кеша, постоянно расширяя структуру TSM. В результате данного исследования рассмотрены основные принципы работы InfluxDB, основные методы и принципы ее работы и дана оценка применимости ее к задуманной системе. Для решения задачи использования прикладного API-интерфейса в вычислениях предложено использовать парадигму GraphQL, позволяющую создать не простое соединение клинт-сервера, а группу микросервисов.

Ключевые слова: база данных, InfluxDB, нейронные сети, репозиторий, GraphQL, анализ, датасет

ANALYSIS OF THE PRINCIPLES AND METHODS OF FORMATION OF TIME-SERIES STORAGE AND ORGANIZATION OF SOFTWARE INTERFACES IN REPOSITO-RIA OF NEURAL NETWORK MODELS

Yamashkin S.A., Skvortsov M.A., Bolshakova M.V., Yamashkin A.A.

Federal State Budgetary Educational Institution of Higher Education «National Research Ogarev Mordovia State University», Saransk, e-mail: dep-general@adm.mrsu.ru

In the Russian Federation, there is an acute issue of strengthening the connectedness of the territory. An important role in solving this issue is played by spatial data infrastructures introduced for processing and analyzing information about complex geosystem models. The core of systems of this class is currently being machine learning algorithms, consolidated in the repositories of neural network models and allowing to automatically identify spatio-temporal patterns and, on the basis of available data, predict the onset or outcome of natural disasters. The purpose of this research is to study in detail the principles of InfluxDB, evaluate its capabilities and consider the possibility of its use as a repository of neural network models. InfluxDB works by using caching technology to store instant results and a WAL restore and startup file that stores the basic structure of the database organization. This file is read only in one case – to restore the system. In order to make the size of this file smaller, the Snappy compression library is applied to it. The TSM columnar storage system is used for disk storage. This format is based on the construction of logarithmically structured merge trees. With this structure, files are processed as the cache is flushed, constantly expanding the TSM structure. As a result of this research, the basic principles of InfluxDB operation, the basic methods and principles of its operation are considered, and an assessment of its applicability to the intended system is given. To solve the problem of using the application API in calculations, it is proposed to use the GraphQL paradigm, which allows you to create not a simple client server connection, but a group of microservices.

Keywords: database, InfluxDB, neural networks, repository, GraphQL, analysis, dataset

В Российской Федерации остро стоит вопрос усиления связанности территории. Немаловажную роль в решении данного вопроса играет эффективное внедрение цифровых инфраструктур пространственных данных (ИПД) регионов России, направленных в основном на оперативную диагностику природно-социально-производственных систем (ПСПС) и высокоточное прогнозирование развития стихийных процессов и явлений. Центральную часть систем данного класса представляют алгоритмы и методы машинного анализа пространственных данных, позволяющие решать целый спектр прикладных задач, таких как обучение признакам, обнаружение аномалий, объединение данных, классификация данных. Предметом анализа могут выступать данные космической съемки, аэрофотосъемка, массивы информации о природных явлениях,

о социальных, экономических и природных объектах, имеющих разрозненную геопространственную организацию. Области применения результатов анализа в народном хозяйстве при этом чрезвычайно широки – от оценки последствий стихийных процессов до повышения эффективности сельского хозяйства [1].

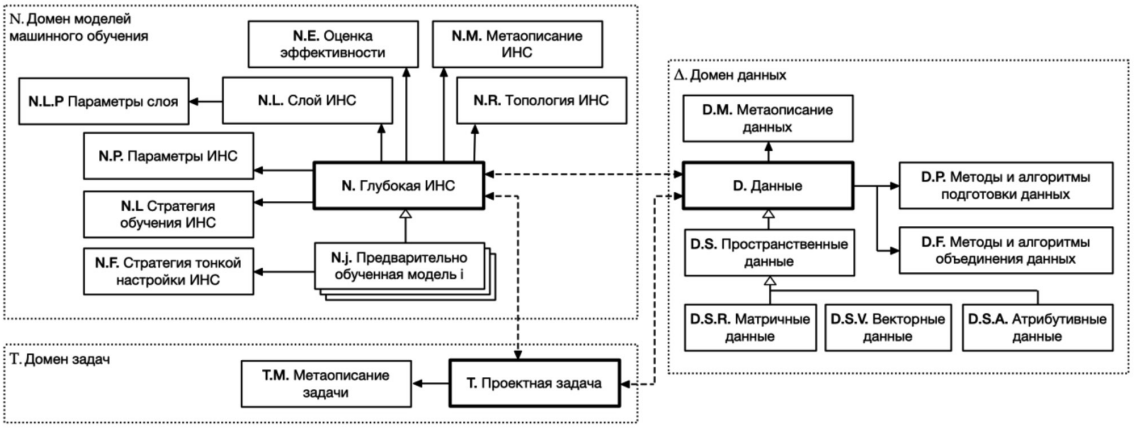
Цель исследования состоит в рассмотрении принципов работы СУБД временных рядов InfluxDB, оценке ее возможностей и характеристик, рассмотрении способов применения InfluxDB, изучении способов интеграции InfluxDB с языком запросов GraphQL для внедрения в репозиторий нейросетевых моделей, а также в рассмотрении возможности ее применения в качестве хранилища для репозитория нейросетевых моделей.

Материалы и методы исследования

Для успешного формирования программной реализации и составления архитектуры проекта разработки репозитория глубоких нейросетевых моделей в целом необходимо продумать и составить онтологическую модель, в которой будет определено формальное описание решаемых задач, алгоритмов обучения и задано отношение между всеми сущностями системы. После анализа всех требований к системе они были закреплены посредством создания онтологической модели. Она определяет требования к хранилищу данных, подразумевает, что все представления разбиты на домены для комплексной формализации изучаемой области знаний – каждая рассматриваемая модель будет соответствовать конкретному набору данных (атрибутивных, растровых, тензорных, векторных) и задач (рисунок).

При помощи такого типа организации репозитория возможно дать формализованное определение исследуемым знаниям, создать базу для проектирования платформенного решения консолидации, а также эффективно использовать и хранить нейросетевые модели для решения проблемно-ориентированных задач [2].

Использование нейросетевых технологий является приоритетной задачей во всем мире. На данный момент уровень их развития поражает воображение. Если раньше они могли отслеживать только статистические данные и выдавать сухие цифры, то сейчас они в реальном времени способны визуализировать свои данные, а их оптимизация достигает такого уровня, что для их работы подойдет любой среднестатистический смартфон. Нейронная сеть (НС) – это математическая модель, которая представляет собой упрощенную в формальном смысле и усложненную в области математической теории модель биологической НС. Обучение НС представляет собой постоянно повторяющийся процесс модификации массива весов посредством существующих математических алгоритмов, целью которых является многомерная оптимизация. При обучении НС используются наборы данных, которые называются датасетами. Датасет – это некоторый упорядоченный набор данных в удобном для применения вместе с нейронной сетью формате, где каждый элемент хранит пару данных – ожидаемый ответ и данные, обрабатываемые НС. По завершении этапа обучения НС мы получаем готовый к использованию набор весов в форме, удобной для восстановления и дальнейшего применения.



Онтологическая модель репозитория для пространственного анализа и прогнозирования с применением глубоких нейронных сетей. Основные сущности

Как правило, при обучении НС записывает полученные веса после каждого шага обучения или после некоторого количества шагов либо лучший результат или согласно любому описанному разработчиком НС алгоритму. Это делается, поскольку существуют риски недоучить или переучить НС. Также это защищает результаты от программных или аппаратных сбоев и позволяет вести статистику обучения с использованием различных датасетов. При данном подходе разработчик может в любой момент вернуться к определенному моменту обучения НС и изменить датасет или значения некоторых весов для получения лучшего результата. В связи с этим разработчики сталкиваются с проблемой хранения и структурирования больших массивов данных. При разработке большого каскада моделей НС предстоит грамотно продумать и автоматизировать данный процесс обработки полученных в результате обучения данных [3].

С.Д. Кузнецов и П.А. Клеменков в статье «Большие данные: современные подходы к хранению и обработке» рассматривали три актуальных подхода к работе с большими данными, а также проанализировали способы решения проблем с обработкой и хранением таких данных [4]. В статье «Big Data – большие данные в бизнесе» автор анализировал различные способы взаимодействия с большими данными, а также рассмотрел использование сервисов для обработки больших данных [5]. Н.С. Паненко и Е.П. Гордиенко в статье «Современные технологии обработки и анализа больших данных в научных исследованиях», в свою очередь, также проанализировали вопрос способа хранения больших данных. Были рассмотрены перспективы развития методов взаимодействия с большими данными в научных исследованиях, а также выполнен обзор технологий хранения больших данных [6].

Результаты исследования и их обсуждение

При перекрестном анализе документации по InfluxDB были выделены основные принципы ее работы, рассмотрены модели ее внутреннего устройства, оценены ее базовые возможности и области применения. Рассмотрим ее более конкретно. Данная БД разработана с целью оптимизации хранения статистических данных. Она достаточно неплохо проявляет себя при внедрении ее в системы интернета вещей, системы отсчета метрик приложений и мониторинга данных. InfluxDB сохраняет значения в виде точек, каждая из которых содержит timestamp, набор полей, набор тегов и измерение.

Для хранения каждой точки используется следующая политика – для точек задаются максимальное время жизни одной точки, количество таких точек в кластере и временной диапазон точек, которые сохранены в данном кластере. Такой подход позволяет проводить быстрый отбор точек, выбирать только заданный временной интервал и проводить над ним любые операции, такие как выборка, удаление, изменение, копирование и др.

Каждый кластер, конечно, может быть отправлен по отдельности, это вызывает большую нагрузку на память и снижение эффективности работы базы данных. Поэтому кластеры принято отправлять группами, называемыми пакетом. Типичный размер пакета – это от сотни до десяти тысяч кластеров.

При создании новых точек InfluxDB производит пересчет существующих точек с целью определения максимально выгодного расположения этих точек, а также проводит их индексирование для обеспечения быстрого доступа к точке и для того, чтобы сделать точку «долговечной». Данная процедура проводится для того, чтобы к точкам можно было легко обратиться в любой момент времени и чтобы была возможность их восстановления после падения сервера с базой данных.

Для обеспечения долговечности точек используется журнал упреждающей (предварительной) записи – Write Ahead Log (WAL). WAL – это файл, в который записывается каждый пакет во время работы системы. Читается данный файл только в одном случае – с целью восстановления системы. Для того чтобы размер этого файла был меньше, к нему применяют библиотеку сжатия Snappy. Snappy – это библиотека, разработанная компанией Google в 2011 г. для быстрой распаковки и сжатия файлов на максимально доступной скорости.

Несмотря на все свои преимущества в области записи данных, система WAL является максимально долгой и сложной для потокового чтения данных. Данная проблема делает такую структуру непригодной для разворачивания на ней системы запросов. Для реализации быстрых и поточных запросов к InfluxDB используется система кеширования. В данном случае в процессе кеширования в оперативную память сервера заносятся структурированные данные, оптимизированные для быстрого извлечения данных. Хранимый кеш держит в себе структуру временных рядов, задан-

ных пользователем, где все поля отсортированы по времени их записи.

Таким образом, две эти структуры – Кеширование и WAL – дополняют друг друга, делая точки долговечными и доступными для запроса. Если в системе произойдет сбой до того, как завершится кеширование, то система восстановит утраченные точки при следующем запуске посредством чтения WAL файла.

Хотя структура WAL и кеширование хорошо работают с входящим потоком временных точек, их недостаточно для долговременного хранения данных в памяти компьютера. Так как файл WAL разворачивается при каждом использовании, необходимо разумно ограничить его размер, чтобы он не отнимал бесконечно долгое время при запуске базы данных и занимал меньшее количество памяти на диске. Что же касается использования кеширования, то данный процесс ограничен размером оперативной памяти и не сможет вместить в себя весь объем данных, хранимых в базе данных. В связи с этим возникает проблема переноса данных в постоянную память, поскольку данные должны быть эффективны по размеру и удобны для организации запросов к ним [7].

Решением данной проблемы служит использование столбчатых методов. Столбчатые методы организуют запись данных по столбцам, а не по строкам, как в классических подходах к хранению данных. Для реализации столбчатого подхода используется формат TSM. Он основан на построении логарифмически структурированных деревьев слияний. При использовании данной структуры файлы обрабатываются по мере сброса кеша, постоянно расширяя структуру TSM. Говоря более простыми словами, в момент предельного заполнения оперативной памяти кешированием оперативная память очищается, передавая свои данные в структуру TSM. После TSM смотрит, есть ли уже записанные данные? Если таковые имеются, то происходят добавление к ним новых данных и уплотнение имеющихся данных для уменьшения размера занимаемой памяти. В противном случае создается первый, базовый кластер данных в памяти, который впоследствии будет расширяться. По мере записывания данных некоторые данные «остывают». Следовательно, их время записи не является приоритетным, и они начинают формировать «холодные» уровни записи данных, к которым больше не применяется процедура уплотнения данных [8].

Программная реализация процедуры уплотнения TSM содержит много высокоуровневой логики и использует запутанные и сложные методы. Но цель, которую она преследует, проста – собрать записываемые кластеры в длинные массивы данных, чтобы максимально улучшить процесс сканирования запросов и сжатия данных.

На основе выбранной БД может быть развернут полномасштабный API-интерфейс для получения данных о конкретной нейронной сети в целях ее использования в дальнейших вычислениях. API (Application programming interface) – это некий программный интерфейс, описывающий способы (т.е. набор функций, классов, констант или структур), благодаря которым одна программа может взаимодействовать с другой, т.е. это некая «прослойка» между приложениями, позволяющая им обмениваться информацией. Данный API-интерфейс будет реализован посредством использования технологии GraphQL. GraphQL – это определенный синтаксис, описывающий, как запрашивать данные с сервера, а также среда выполнения таких запросов. Преимущества данной технологии заключаются в следующем: GraphQL позволяет пользователю указать, какие именно данные ему нужны, не перезагружая запрос лишней информацией; облегчает объединение данных из разных источников. В API-интерфейсе будет реализовано получение максимально обученных весов на текущий момент времени для выбранной модели обучения в целях использования совместно с репозиторием моделей машинного обучения.

Выводы

Проведенное исследование позволяет сделать следующие выводы.

1. Рассмотренная в статье СУБД InfluxDB не является единственным решением, использование которого необходимо и достаточно для функционирования репозитория нейросетевых моделей: для построения систем данного класса целесообразно применение мультимодельного подхода к организации хранилища. Одновременно с этим база данных временных рядов позволяет вести временную статистику обучения, отслеживать прогресс и сохранять обученные веса в процессе прохождения каждой эпохи. При правильной оптимизации сохранений в базу данных мы сможем отслеживать статистику обучения нейронной сети и возвращаться к удачным шагам, вручную меняя конфигурацию сети, и продолжать более корректное ее обучение. Использование данного подхода позволяет максимально упростить работу в высоконагруженных сетях.

2. Для обеспечения целостности хранимых данных необходимо использование журнала упреждающей записи (WAL) – файла, в который записывается каждый пакет во время работы системы. Однако для реализации быстрых и поточных запросов к InfluxDB целесообразно применение системы кеширования, которая заносит в оперативную память сервера структурированные данные, оптимизированные для быстрого извлечения данных. Наконец, решением проблемы переноса данных в постоянную память выступает использование столбчатых методов с применением формата TSM, основанного на построении логарифмически структурированных деревьев слияний.

3. Для решения задачи развертывания прикладного программного API-интерфейса для получения данных о конкретной нейронной сети в целях использования ее в дальнейших вычислениях предложено применять парадигму GraphQL, позволяющую создать не простое соединение клинт-сервера, а группу микросервисов. Микросервисная архитектура – это набор небольших сервисов, слабо связанных между собой. Это позволяет обеспечить большую масштабируемость, более точную настройку под запросы пользователей и снизить нагрузку на глобальные узлы нашей системы.

4. При организации микросервисной архитектуры появляется возможность создать собственную базу данных под каждую модель нейронной сети, тем самым снизив нагрузку на одну отдельно взятую базу данных, исключив риск падения всей системы в случае неполадки определенной модели, и правильно распределить поток юзеров между базами данных. GraphQL способен организовать бесперебойный доступ к существующим сервисам. Удобство применения данной технологии состоит в том, что после настройки GraphQL весь набор микросервисов выступает как единое целое, GraphQL полностью сам регулирует поток пользователей и вызывает запросы к определенной базе данных на основании параметров запроса. Тем самым мы получаем полностью автоматизированную систему, не требующую от разработчиков написания дополнительных интерфейсов при взаимодействии между микросервисами и интерфейсным приложением.

Работа выполнена при финансовой поддержке гранта Президента Российской Федерации (грант № МК-199.2021.1.6).

Список литературы

1. Ямашкин С.А., Ямашкин А.А., Занозин В.В. Формирование репозитория глубоких нейронных сетей в системе цифровой инфраструктуры пространственных данных // Потенциал интеллектуально одарённой молодежи – развитию науки и образования: материалы IX Международного научного форума молодых ученых, инноваторов, студентов и школьников. 2020. С. 370–375.

2. Скворцов М.А., Большакова М.В., Ямашкин С.А., Ямашкин А.А. Сравнительный анализ подходов к управлению базами данных для организации хранилища репозитория нейросетевых моделей // Современные наукоемкие технологии. 2021. № 6 (1). С. 108–113.

3. Федюшкин Н.А., Ямашкин С.А. Исследование репозиториев моделей нейронных сетей // Научно-технический вестник Поволжья. 2021. № 3. С. 28–30.

4. Клеменков П.А., Кузнецов С.Д. Большие данные: современные подходы к хранению и обработке // Труды Института системного программирования РАН. 2012. Т. 23.

5. Сизов И.А. Big Data-большие данные в бизнесе // Экономика. Бизнес. Информатика. 2016. Т. 2. № 3. С. 8–23.

6. Гордиенко Е.П., Паненко Н.С. Современные технологии обработки и анализа больших данных в научных исследованиях // Актуальные проблемы железнодорожного транспорта. 2018. С. 44–48.

7. Halpin T. Object-role modeling (ORM/NIAM). Handbook on architectures of information systems. Springer, Berlin, Heidelberg, 1998. P. 81–103.

8. Miller J.J. Graph database applications and concepts with Neo4j. Proceedings of the Southern Association for Information Systems Conference, Atlanta, GA, USA. 2013. P. 141–147.