Софийски университет „Св. Климент Охридски”
Факултет по математика и информатика,
катедра “Софтуерни технологии”

Методи за реализация на софтуерни системи за обработка на големи данни

Симеон Стоичков Емануилов

Автореферат

на дисертация
за присъждане на образователна и научна степен “доктор” в професионално направление 4.6 Информатика и компютърни науки, докторска програма „Софтуерни технологии” - Софтуерно инженерство

Научен ръководител: доц. д-р Александър Димов

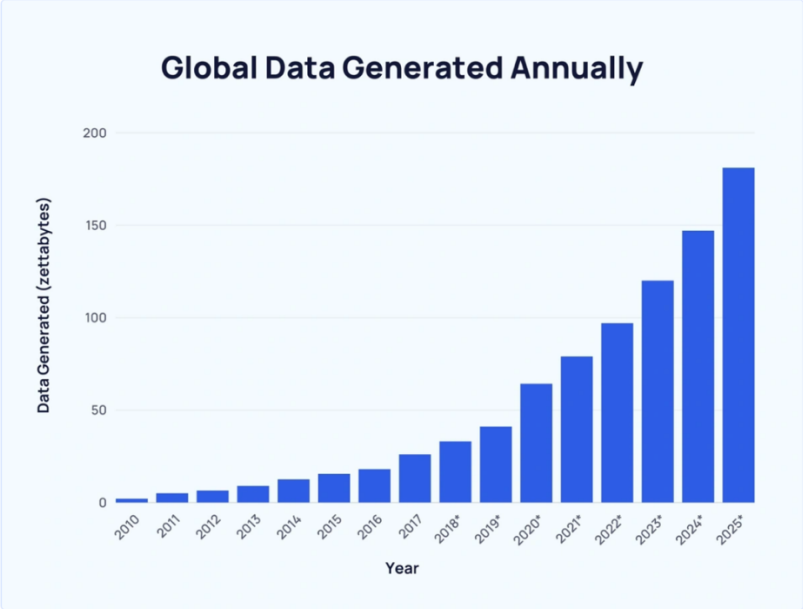
София
2025

Въведение

Актуалност на изследването

Бурният растеж в обема на данните през последните години, който се дължи на технологичния напредък, разпространението на устройства, генериращи данни, и все по-широкото прилагане на приложения с интензивно използване на информация, се превърна в критично предизвикателство в различни сектори [1].

Това явление, често наричано „експлозия на данните“, има дълбоки последици за бизнеса, научните изследвания и обществото като цяло.



Изображение 1. Обем на данни, създадени, копирани и използвани в световен мащаб от 2010 до 2020 г., с прогнози за периода 2021-2025 г.[1]

Интернет на нещата (IoT) и операциите на предприятията са основните фактори, които допринасят за това нарастване на данните [2]. IoT устройствата събират и предават значителни информационни потоци, което налага разработването на надеждни ИТ системи за тяхното ефективно управление и анализ. Едновременно с това, развитието на платформите за обработка на големи данни засилва необходимостта от усъвършенствани инструменти за изследване и визуализация, целящи извличането на стойност от комплексните информационни масиви.

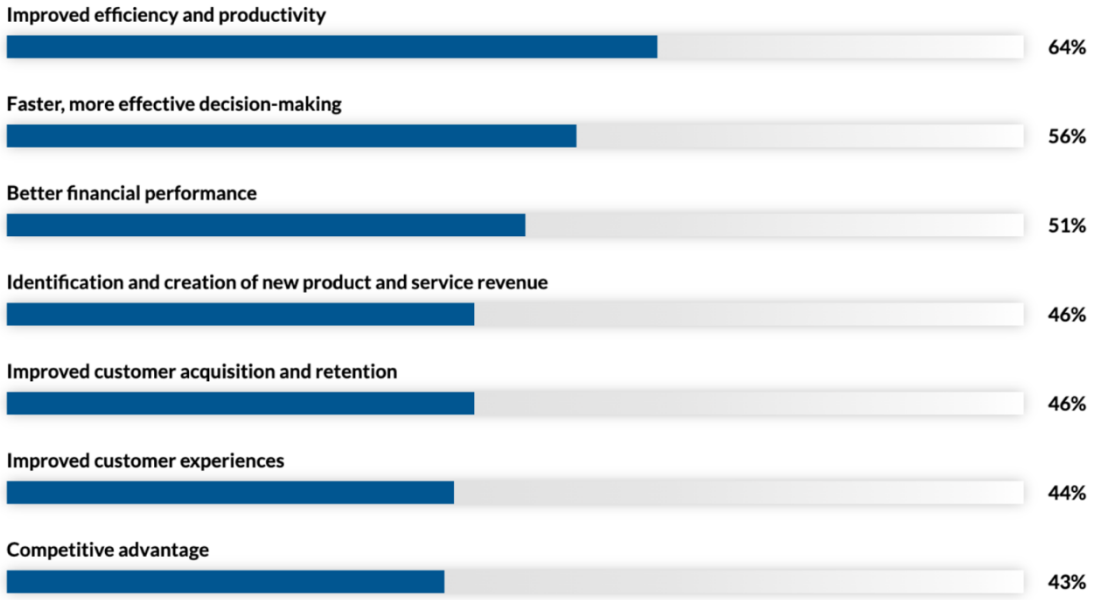
Наред с това, системите за машинно самообучение (МС) функционират както като значителни потребители, така и като генератори на данни. Обучението на МС модели изисква наличието на обширни информационни масиви, като впоследствие самите модели генерират значителни обеми изходна информация. Тази тясна взаимовръзка между МС и големите данни допълнително подчертава необходимостта от ефективни решения за тяхната обработка и съхранение [3].

Последиците от този нарастващ обем данни са действително широкообхватни. Организациите, които внедряват адекватни технологии за тяхното управление и анализ

[1] Ръст на данните в световен мащаб 2010-2025 от Statista:
https://www.statista.com/statistics/871513/worldwide-data-created/

(Изображение 2), получават възможност да повишат своята оперативна ефективност и да оптимизират процесите по вземане на решения.

Същевременно обаче, тези организации се сблъскват и със значителни предизвикателства в сферата на управлението на данните, особено по отношение на мащабируемостта, съхранението и ефективността на тяхната обработка. Въпреки тези трудности, способността за анализ на подобни информационни обеми се утвърждава като ключово конкурентно предимство, което разкрива нови пазарни възможности и стимулира иновациите в редица отрасли и научни сфери.



Изображение 2. Ползи за компаниите от използването на анализи[2]

С непрекъснатото нарастване на обема на данните (Изображение 1) разработването на нови архитектурни подходи, иновативни техники за представяне на информация и усъвършенствани методологии за нейната обработка придобива все по-голямо значение [4]. Тези постижения са ключови както за ефективното управление на съвременните информационни потоци, така и за пълноценното оползотворяване на потенциала на технологиите, базирани на данни, в разнообразни области като здравеопазване, финанси, научни изследвания и други значими сектори.

Настоящият дисертационен труд е насочен към адресиране на неотложните предизвикателства, присъщи на системите с интензивно използване на данни, чрез изследване и разработване на решения за повишаване на ефективността, мащабируемостта и интерпретируемостта при обработката на широкомащабни информационни масиви.

Обект

Обект на изследване са софтуерните системи, предназначени за интензивна обработка на данни, както и ключовите аспекти на тяхното проектиране, реализация и

[2] Columbus, Louis. The Global State Of Enterprise Analytics, 2020 (Глобалното състояние на корпоративните анализи, 2020 г.): https://www.forbes.com/sites/louiscolumbus/2019/10/21/the-global-state-of-enterprise-analytics-2020/

функциониране. Тези системи се характеризират със способността си да обработват, трансформират и анализират огромни информационни масиви – потенциално възлизащи на милиарди записи – като същевременно са изправени пред специфични предизвикателства, свързани с производителността, мащабируемостта и адекватното представяне на данните [5].

Предмет

Предмет на изследването е разработването, оптимизацията и оценката на иновативни методи и техники, насочени към реализацията, управлението и ефективната експлоатация на софтуерни системи, предназначени за интензивна обработка на данни.

Цел

Основната цел на настоящия дисертационен труд е да допринесе за усъвършенстването на методите за реализация на софтуерни системи, работещи с интензивни потоци от данни. Това се постига чрез разработване и оценка на нови подходи за управление и анализ на мащабни информационни колекции, идентифициране на ограниченията в съществуващите практики и предлагане на иновативни решения, насочени към повишаване на мащабируемостта, производителността и интерпретируемостта на тези системи.

Изследователският фокус е насочен към оптимизирането на техниките за представяне, съхранение, индексиране и извличане на данни. Специално внимание е отделено на колонно-ориентираните бази данни, векторните представяния и алгоритмите за търсене по сходство, приложими към информационни масиви от порядъка на милиарди записи.

Наред с това, дисертационният труд цели да подобри интерпретируемостта на комплексните структури от данни. За постигането на тази цел се изследват техники за трансформиране на високоизмерни векторни представяния в по-интуитивни и лесно разбираеми за човека форми.

Задачи

За постигане на целта на дисертационния труд бяха формулирани следните основни задачи:

Задача 1: Анализ и оценка на архитектурни подходи за системи, работещи с интензивни потоци от данни, по отношение на тяхната ефективност за справяне с предизвикателствата, свързани с мащабируемостта и производителността при обработка на големи информационни обеми.

Задача 2: Проучване и разработване на усъвършенствани техники за индексиране и търсене на високоизмерни данни в информационни масиви от порядъка на милиарди записи, с фокус върху процесорно-базирани решения за мащабируемо търсене по сходство[3].

Задача 3: Изследване на методи за подобряване на интерпретируемостта на комплексни представяния в системите с интензивно използване на данни, с акцент върху

[3] Анализ на разходите за GPU спрямо CPU при споделен хостинг: https://medium.com/@GPUnet/gpu-vs-cpu-cost-analysis-in-shared-hosting-environments-bda82a65a1df

трансформирането на високоизмерни цифрови вектори в по-лесно разбираеми и интуитивни форми.

Задача 4: Изследване и разработване на оптимизирани модели на данни за ефективно съхранение, извличане и обработка в контекста на системи с интензивно използване на данни, с особен акцент върху архитектури, обработващи високоскоростни потоци от данни и аналитични задачи.

Задача 5: Провеждане на експериментални изследвания и сравнителен анализ за оценка на ефективността на предложените подходи в практически сценарии, включващи мащабно търсене по сходство и интерпретация на данни.

Методология

За решаването на изследователските задачи и постигането на целите на дисертационния труд се прилага комбиниран изследователски подход, включващ теоретичен анализ, проектиране и разработка на софтуерни системи, и емпирична оценка. Основните компоненти на използваната методология са:

1) Обзор на съществуващата литература

Извършен е подробен преглед и критичен анализ на актуалното състояние на изследванията в областта на системите с интензивно използване на данни. Разгледани са релевантни архитектурни модели и утвърдени техники за представяне на информация. Основната цел на този обзор е идентифицирането на съществуващи ограничения и очертаващи се насоки за развитие, които служат като основа при дефинирането на изследователските направления в дисертацията. Литературният преглед обхваща следните ключови области: архитектурни стилове, модели на данни, методи за представяне на данни и нововъзникващи технологични тенденции.

2) Разработка на системи и библиотеки

Този етап от методологията включва проектирането и практическата реализация на софтуерни системи и библиотеки. Тези артефакти са предназначени да демонстрират конкретното приложение и предимствата на предложените в дисертационния труд методи. Разработените системи и библиотеки служат като основа за последващата експериментална оценка на техните характеристики и ефективност.

3) Емпирична оценка

Ключов компонент на изследователската работа е провеждането на емпирична оценка на предложените методи и разработените софтуерни решения. Тази оценка се осъществява чрез експерименти с реални или представителни набори от данни, с цел обективно измерване и анализ на постигнатата производителност, мащабируемост и други релевантни показатели.

Структура

Дисертационният труд е структуриран в увод, шест основни глави, заключение, раздел с научни и научно-приложни приноси, библиография и приложения.

В увода е представена актуалността на изследвания проблем, дефинирани са обектът, предметът и целта на дисертационния труд, формулирани са основните задачи и е описана използваната методология. Обсъдени са също така приложимостта и очакваните ползи от изследването.

Основното изложение е разгърнато в шест глави със следното съдържание:

Първа глава полага основите на дисертационния труд, като извършва преглед на системите с интензивно използване на данни и идентифицира свързаните с тях предизвикателства.

Втора глава разглежда техниките за индексиране и клъстеризация, които са от съществено значение за управлението и търсенето в големи информационни масиви.

Трета глава е посветена на предизвикателствата при търсене по сходство в милиарден мащаб, като представя и обосновава нов хибриден алгоритъм за индексиране.

Четвърта глава дискутира методи за подобряване на интерпретируемостта на плътни векторни представяния и представя LangVec [6] – нова софтуерна рамка, разработена за тази цел.

Пета глава анализира колонно-ориентираните модели на данни и тяхното приложение в контекста на системи с интензивно използване на данни.

Шеста глава представя резултатите от проведените експериментални изследвания и сравнителни анализи, които валидират ефективността на предложените в труда методи.

В **заключението** са обобщени основните научни констатации и резултати, синтезирани са изводите от всички глави, обсъдени са импликациите за развитието на софтуерните системи с интензивно използване на данни и са очертани бъдещи насоки за изследователска и развойна дейност.

Разделът „**Приноси**” детайлизира научните и научно-приложните приноси на дисертацията.

Представен е **списък на научните публикации** по темата на дисертацията, описващ статиите, публикувани или приети за печат в рецензирани списания и сборници от конференции.

Включен е и **списък на участията в научни конференции**, на които са представени и обсъдени резултати от проведеното изследване.

Библиографията съдържа списък на всички цитирани в дисертационния труд източници, оформен съгласно стандарта APA. Бележките под линия са използвани за допълнителни пояснения, когато е необходимо.

Глава 1: Контекст, основни понятия и литературен обзор

Настоящата глава представя преглед на ключовите концепции, актуалното състояние на научните изследвания и нововъзникващите тенденции в областта на системите с интензивно използване на данни. Целта на извършения литературен обзор е да се положи солидна теоретична основа за изследванията, изложени в следващите глави, както и да се идентифицират съществуващи пропуски в научните познания, към чието преодоляване е насочен настоящият дисертационен труд.

1.1 Въведение в системите с интензивно използване на данни

През последните години широкото навлизане на цифровите технологии и експоненциалното нарастване на обема на генерираните данни доведоха до формирането на нов клас софтуерни системи, които се характеризират с интензивна работа с данни (data-intensive systems). Ключова особеност на тези системи е, че основното предизвикателство при тях е ефективното управление, обработка и анализ на големи обеми информация, а не толкова изчислителната сложност на алгоритмите [7]. Системите от този тип трябва да се справят не само с огромното количество данни, но и с тяхната нарастваща комплексност и високата скорост на изменение и генериране [8].

Концепцията за системите, работещи интензивно с данни, е тясно свързана с по-широката област на „големите данни“ (Big Data), които традиционно се дефинират посредством модела на трите „V“ (3Vs): обем (Volume), скорост (Velocity) и разнообразие (Variety) [7]. Впоследствие някои изследователи разширяват този модел до четири или пет „V“-та, добавяйки характеристики като достоверност (Veracity) и стойност (Value). Системите, базирани на интензивна обработка на данни, намират приложение в широк спектър от области – от електронна търговия и пазарно разузнаване до научни изследвания и публична администрация [7].

Системите, характеризиращи се с интензивна обработка на данни, интегрират разнообразни компоненти и технологии, сред които фигурират бази данни, кеширащи механизми, индекси за търсене, системи за поточна обработка (stream processing) и такива за пакетна обработка (batch processing) [8]. Адекватният избор и ефективната интеграция на тези компоненти са от ключово значение за постигане на желаната производителност и мащабируемост на системата. За разлика от традиционните софтуерни архитектури, приложенията, работещи интензивно с данни, често налагат използването на специализирани архитектурни подходи с цел успешното преодоляване на специфичните предизвикателства, произтичащи от управлението на огромни информационни обеми [9].

Разработката на подобни системи неизменно изисква прилагането на мултидисциплинарен подход и тясно сътрудничество между експерти от съответната предметна област, софтуерни инженери, специалисти по анализ на данни (data scientists) и експерти по облачни изчислителни технологии [10].

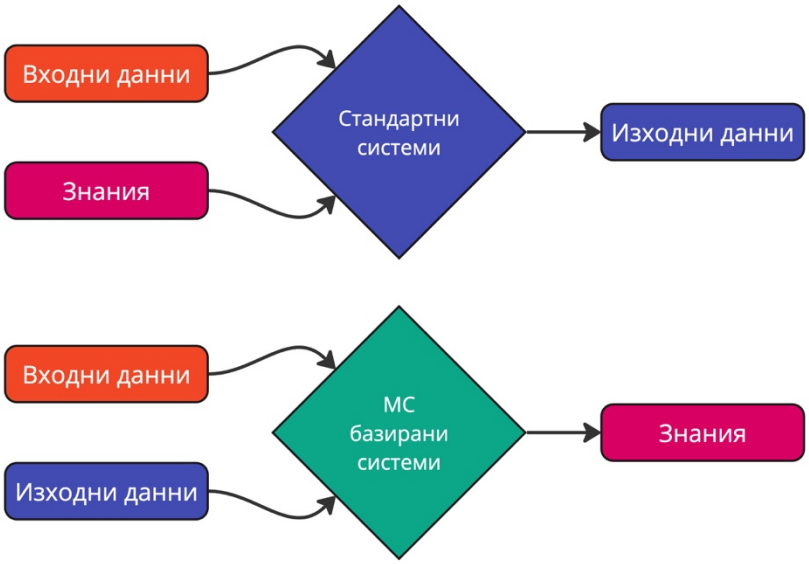
В обобщение, системите с интензивно използване на данни бележат значителна трансформация в областта на софтуерното инженерство, обусловена от непрекъснато нарастващия обем и стратегическата важност на данните в съвременните приложения. Задълбоченото познаване на специфичните им характеристики, присъщите предизвикателства и адекватните архитектурни подходи е фундаментално условие за разработването на ефективни и устойчиви решения в настоящата епоха на големите данни.

1.2 Системи с елементи на машинно самообучение спрямо традиционни системи

През последните години се отчита значителен ръст в разработката и приложението на системи, интегриращи елементи на машинно самообучение (МС). Това обуславя разграничаването на два основни типа системи: такива, базирани на МС, и традиционни софтуерни системи. Системите с МС могат да бъдат дефинирани като програмни продукти, които включват поне един компонент за машинно самообучение, способен да се учи и адаптира въз основа на данни, прилагайки утвърдени алгоритмични подходи.

Традиционните софтуерни системи, от своя страна, са предимно детерминистични. Те се характеризират с предварително дефинирани правила и строго структурирани процеси за обработка на данните [11]. При тях разработчиците изрично програмират заложената бизнес логика, което осигурява предвидимост на резултатите при подаване на конкретни входни данни. Подобни системи обикновено следват „отгоре-надолу“ (top-down) подход на разработка и демонстрират консистентна производителност в рамките на предварително зададени оперативни параметри.

В ярък контраст, системите, базирани на МС, притежават способността да извличат закономерности (модели) и да вземат решения въз основа на анализ на данни, без да е необходимо изричното програмиране на всички правила за поведение (Изображение 3).



Изображение 3. Сравнение на традиционните процеси на разработване на софтуер и на машинното обучение

Както е илюстрирано на изображение 3, традиционните системи функционират въз основа на изрично дефинирана логика, докато системите, базирани на МС, извличат входно-изходни зависимости директно от данните, адаптирайки се с течение на времето.

Тази адаптивност е ключова характеристика на изкуствения интелект (ИИ), в рамките на който машинното самообучение представлява основно подмножество, а дълбокото обучение (ДО) – негова по-нататъшна специализация. Въпреки че в настоящия дисертационен труд терминът „системи, базирани на МС“ се употребява в широк смисъл, изследването е фокусирано предимно върху изчислително интензивни системи, които често, макар и не изключително, прилагат техники на дълбокото обучение.

Тези фундаментални различия пораждат специфични предизвикателства в процесите на проектиране, реализация и оптимизация на МС системите [11], [12], [13], [14]. Макар традиционните принципи на софтуерното инженерство да остават приложими за определени аспекти на системите с МС, пренебрегването на техния подчертано ориентиран към данни характер и тяхната вероятностна същност може да доведе до неоптимално системно проектиране и последващо внедряване [15].

Обширно проучване на Serban и др. [15] анализира практиките на софтуерното инженерство в пълния жизнен цикъл на разработка на МС системи, базирано на анкета сред над 400 специалисти. Техните резултати показват, че въпреки тенденцията в по-големите организации да се прилагат по-всеобхватни практики, аспектите, свързани с надеждността на системите, често биват недостатъчно адресирани. Изследването също така осветлява връзката между различните инженерни практики и ключови показатели за екипна ефективност като гъвкавост и отчетност.

Въз основа на тези наблюдения, Sculley и др. [16] въвеждат концепцията за „технически дълг“ в системите с машинно самообучение. Специално внимание е обърнато на аспектите на тестване и валидиране на МС системи в изследването на Breck и др. [17] , където е разработен и представен цялостен набор от двадесет и осем добри практики, покриващи пълния жизнен цикъл на разработка. Трудът на Breck и др. допринася съществено за разбирането на влиянието на тестовите практики върху надеждността на системата и дългосрочните разходи за нейната поддръжка.

Индустриалната перспектива е представена в проучвания като това на Amershi и др. [18] от Microsoft, което документира инженерните практики при разработката на МС системи в множество екипи. Макар техните резултати да предлагат ценни сведения за развитието на МС в голям корпоративен мащаб, по-широката им приложимост към по-малки организации остава отворена област за бъдещи изследвания.

Тези научни разработки полагат основата за по-добро разбиране на предизвикателствата и специфичните изисквания при инженерната разработка на системи с МС, като същевременно очертават области, нуждаещи се от последващи задълбочени изследвания.

1.3 Колонно-ориентирани модели на данни: Принципи и отличителни характеристики

Колонно-ориентираните бази данни (column-oriented databases) организират съхранението на информацията по колони, за разлика от традиционното редово подреждане. Тази фундаментална промяна в архитектурата на съхранение води до редица съществени предимства, особено за системи, които обработват големи обеми данни и изпълняват често аналитични заявки.

Именно основополагащите принципи на колонно-ориентираните модели ги правят изключително подходящи за системи, работещи интензивно с данни. Такива системи обикновено дават приоритет на аналитичната обработка и изискват високопроизводително изпълнение на заявки върху големи информационни масиви.

1.4 Предизвикателства при представянето и интерпретацията на данните

Системите, характеризиращи се с интензивно използване на данни, се сблъскват с множество предизвикателства при представянето и интерпретацията на огромни, комплексни и често високоизмерни информационни масиви. Тези предизвикателства

засягат разнообразни аспекти на управлението на данните – от тяхното съхранение и извличане до последващия анализ и ефективна визуализация. Научната литература по темата идентифицира няколко основни групи проблеми в това направление:

1) **Голяма размерност и „проклятието на размерността“**

С нарастването на броя на измеренията (размерността) на данните, традиционните структури и алгоритми за търсене по сходство губят значителна част от своята ефективност [3].

2) **Мащабируемост и производителност**

Предвид експоненциалното нарастване на обемите от данни, разгледано във въведението, осигуряването на мащабируема и високопроизводителна обработка на информация се превръща във все по-значимо предизвикателство.

3) **Тълкуване на сложни представяния на данни**

С увеличаването на сложността на моделите за представяне на данни, интерпретирането и визуализацията на високоизмерни данни стават все по-трудни задачи за анализаторите [19].

4) **Ефективно филтриране и многомерни заявки**

Много приложения, работещи интензивно с данни, изискват способността за ефективно изпълнение на комплексни многомерни заявки. Съществуващите традиционни структури за индексиране обаче често изпитват затруднения при поддържането на такива заявки в голям мащаб [8].

5) **Балансиране на използването на паметта и производителността на заявките**

Постигането на висока производителност при изпълнение на заявки често е обвързано със съхраняването на значителни обеми данни в оперативната памет. С нарастването на размера на информационните масиви обаче този подход става все по-ресурсоемък и непрактичен.

Основна цел на научните изследвания в тази област е преодоляването на ограниченията на традиционните подходи, което да позволи по-ефективно управление и интерпретация на данните в широкомащабните системи с интензивно използване на данни.

1.5 Обобщение

В настоящата глава бяха очертани основните концепции, свързани със системите, работещи интензивно с данни, и бяха разгледани критичните предизвикателства при представянето и интерпретацията на информацията в тях. Беше изтъкнато, че „проклятието на размерността“ продължава да затруднява ефективната обработка на данни в многомерни пространства, а въпросите, свързани с мащабируемостта и производителността, остават константни проблеми за широкомащабните системи.

В обобщение, направеният в главата анализ полага основите за разбиране на съвременния контекст и спецификите на архитектурното проектиране на системи с интензивно използване на данни. Идентифицирани са ключови области, в които иновационните решения са от съществено значение за посрещане на високите изисквания за мащабируемост и производителност, характерни за съвременните интензивни задачи. Изводите от този преглед служат като отправна точка за

разработването на нови методи – включително иновативна техника за хибридно индексиране и други оптимизационни подходи – които са предмет на разглеждане в следващите глави.

Следващата глава адресира пряко първото от тези фундаментални предизвикателства. В нея се извършва анализ на съществуващите техники за индексиране и клъстеризация, които са от съществено значение за ефективното управление и търсене в големи информационни масиви.

Глава 2: Техники за индексиране и клъстеризация на широкомащабни данни

Както беше изтъкнато в предходната глава, ефективното управление и търсене на широкомащабни, високоизмерни данни поставят множество предизвикателства пред съвременните системи, характеризиращи се с интензивна обработка на данни.

Ефективните техники за индексиране и клъстеризация са от критично значение за успешното управление и търсене в големи информационни масиви. В контекста на високоизмерните данни, една от основните задачи при търсене е идентификацията на сходни елементи. С нарастването на обема на данните до милиарди записи, сложността при извършване на такова търсене по сходство върху високоизмерни вектори се увеличава значително [20].

2.1 Високоизмерни данни и търсене по сходство

Високоизмерните данни представляват набори от данни, при които множество характеристики (или променливи) описват всяка точка от данните. Така например, данни се считат за високоизмерни, ако в даден набор от тях всеки обект (например къща) е представен чрез стотици характеристики (като местоположение, площ, възраст и др.). Такива данни често са резултат от инженерство на характеристиките (feature engineering), което се основава в голяма степен на техники за векторизация за преобразуване на първичните (суровите) данни в числови векторни представяния [11].

В много приложения за машинно самообучение, като например семантично търсене и системи за препоръки, плътните векторни представяния се използват за кодиране на информация с цел нейното ефективно и точно извличане [21]. Тези плътни вектори са компактни числови представяния с фиксирана дължина, които отразяват ключовите характеристики и зависимости на входните данни [6].

Високата размерност на тези вектори и техният числов характер обаче усложняват тяхната интерпретация. Съществуващите техники за интерпретация на вектори, като намаляване на размерността и методите за визуализация [22], [23], [24], често не предоставят пряка и интуитивна интерпретация на отделните вектори. Повече подробности относно интерпретацията са разгледани в Глава 4.

Търсенето по сходство е основна операция в множество приложения, които работят с високоизмерни данни [25], [21]. Неговата цел е да идентифицира точки от данните, които са близки до дадена заявка (представена като точка или вектор), въз основа на определена метрика за разстояние (напр. евклидово разстояние) или сходство (напр. косинусово сходство).

2.2 Системи за семантично търсене

Системите за семантично търсене бележат напредък в извличането на информация, надхвърляйки традиционното съпоставяне по ключови думи чрез по-дълбоко разбиране на намеренията на потребителя и контекстуалното значение. Тези системи използват онтологии, семантични анотации и усъвършенствани техники за логически извод с цел повишаване на точността на резултатите от търсенето.

В основата на семантичното търсене лежат онтологиите, които осигуряват структурирана рамка за представяне на знания, специфични за дадена област. Архитектурата на тези системи често включва графово-структурирани данни, както е

подчертано от Tran и съавтори [26], което позволява навигация по сложни взаимовръзки между елементите от данните. В сходен подход, Gärtner и съавтори [27] предлагат интерактивни, онтологично-базирани механизми за формулиране на заявки. Тези механизми дават възможност на потребителите да прецизират своите стратегии за търсене, като се съобразяват със семантичната структура на базовите данни.

Основната цел на системите за семантично търсене е да повишат точността на извличаната информация чрез по-добро разбиране на потребителските цели и контекстуалното значение на заявките [28]. Те представляват значителен напредък спрямо традиционните методи за търсене, тъй като отчитат сложността на потребителските намерения и взаимовръзките между данните в широкомащабни хранилища.

Този напредък в областта на системите за семантично търсене създава предпоставки за разработването на по-усъвършенствани подходи за управление и анализ на големи информационни масиви.

2.3 Предизвикателства при мащабно търсене по сходство

Разработването на широкомащабни системи за търсене по сходство е съпроводено с редица съществени предизвикателства, които стават особено изразени при нарастване на обема на данните до милиарди записи. Тези предизвикателства произтичат от присъщата сложност на високоизмерните данни (разгледана в раздел 2.1), изчислителните ограничения и необходимостта от ефективни механизми за съхранение и извличане на информация.

В следващите подраздели са описани основните предизвикателства, пред които е изправено търсенето по сходство в милиарден мащаб.

2.3.1 Проклятието на размерността

„Проклятието на размерността“, разгледано в Глава 1 (раздел 1.4.1), представлява фундаментално математическо и изчислително предизвикателство при широкомащабното търсене по сходство [29].

2.3.2 Мащабируемост и производителност

С разширяването на мащаба на информационните масиви до милиарди записи, производителността при изпълнение на заявки се сблъсква с фундаментални изчислителни бариери, чиято сложност нараства нелинейно с обема на данните [21]. Тази зависимост може да бъде илюстрирана количествено: докато заявка към набор от данни с милиони записи може да се изпълни за милисекунди, обработката на същата заявка към набор от данни с милиарди записи не просто отнема хилядократно повече време – влошаването на производителността често е значително по-сериозно. Това се дължи на съвкупното действие на фактори като ограничения в йерархията на паметта, тесни места във входно-изходните операции (I/O) и нарастващата сложност на индексирането.

Тези ограничения, особено по отношение на паметта, представляват съществена пречка пред разработването на решения за мащабна обработка и извличане на данни. Преодоляването на тези предизвикателства пред мащабируемостта е от ключово значение за разработването на системи, способни ефективно да управляват и анализират информационни масиви от порядъка на милиарди записи.

2.3.3 Тесни места в съхранението и входно-изходните (I/O) операции

Управлението на информационни масиви, чийто обем надхвърля наличния капацитет на оперативната памет, поражда допълнителни усложнения по отношение на ефективността на тяхното съхранение и извличане. Входно-изходните операции с дискови устройства (disk I/O) се превръщат в критично ограничение за производителността при работа с такива големи информационни масиви, което изисква прецизна оптимизация на стратегиите за зареждане и кеширане на данни [30]. Това предизвикателство е особено значимо при сценарии, изискващи интензивен достъп до дисковите устройства за обработка на заявки или за изпълнение на задачи за анализ.

2.3.4 Многомерно филтриране

Включването на сложни критерии за филтриране в процеса на търсене по сходство въвежда допълнителни нива на сложност в архитектурата на системата. Ефективното интегриране на векторното търсене по сходство с филтриране, базирано на атрибути, представлява значително техническо предизвикателство, особено при работа с многомерни филтри [31].

2.3.5 Специфични за хардуера ограничения

Въпреки че подходите, ускорявани от графични процесори (GPU), демонстрират значителен потенциал за търсене по сходство в милиардни мащаби [21], тяхното широко разпространение е свързано с определени ограничения. Разходите за високопроизводителен графичен хардуер (GPU) представляват съществена пречка, особено в среди с ограничени ресурси. Тази дихотомия между подходите, базирани на графични (GPU) и централни (CPU) процесори, подчертава необходимостта от внимателна оценка на хардуерните изисквания в контекста на конкретните сценарии на употреба и наличните ресурси.

2.3.6 Компромис между точност и ефективност

Алгоритмите за приблизително търсене на най-близък съсед (ANN) са насочени към постигане на оптимален баланс между точността на търсене и изчислителната ефективност [32], [33]. Постигането на това равновесие обаче става все по-трудно с нарастването на обема на данните до милиарди записи. Основният компромис между пълнотата (recall) и времето за изпълнение на заявка става все по-критичен при такъв мащаб. В резултат, предизвикателството да се поддържа висока пълнота, едновременно с осигуряването на ниска латентност, нараства пропорционално на обема и размерността на данните.

2.3.7 Разпределение и хетерогенност на данните

Информационните масиви в реални условия често се характеризират с неравномерни разпределения. Това свойство може значително да повлияе на ефективността на методите за индексиране, базирани на разделяне на пространството. Подобни неравномерни разпределения поставят под съмнение основополагащите допускания на много традиционни техники за индексиране, което налага разработването на алтернативни подходи. Тези подходи трябва да са способни да се адаптират към вариациите в гъстотата на данните и наличието на клъстери във високоизмерното пространство.

В следващата секция ще бъдат разгледани съществуващите подходи за индексиране, а в глава 3 ще бъде представен нов хибриден метод, предназначен за справяне с тези предизвикателства в контекста на търсенето по сходство в милиарден мащаб.

Целта на настоящия труд в този контекст е да допринесе за по-доброто разбиране на съвременното състояние на изследванията в областта, както и да очертае нови насоки за напредък в сферата на мащабното индексиране и извличане на данни.

2.4 Преглед на съществуващите подходи за индексиране

Един от основните подходи в тази област е IVF (Inverted File Index – обърнат файлов индекс), предложен от Jégou и съавтори [32]. IVF разделя пространството на търсене на по-малки области, за което обикновено се прилагат техники за клъстеризация, например k-means. Това разделяне позволява по-ефективно търсене, като ограничава изследването до най-перспективните области на пространството.

Въз основа на концепцията за IVF са разработени техники за продуктово квантуване (Product Quantization, PQ). Те допълнително компресират високоразмерни вектори, като същевременно запазват информацията за сходство между тях. PQ методите разделят векторите на подвектори и квантуват всеки подвектор независимо, което осигурява компактно представяне и ефективно изчисляване на разстоянията [34]. Този подход е показал висока ефективност при широкомащабни приложения, където ефективността на съхранение и изчислителната производителност са от критично значение.

Графово-базираните методи представляват друг значителен клон на подходите за индексиране. Алгоритъмът HNSW (Hierarchical Navigable Small World graphs), предложен от Малков и Яшунин [33], се утвърди благодарение на добрите си показатели както по отношение на точността, така и на скоростта на изпълнение на заявките. HNSW конструира многослойна графова структура, която позволява ефективна навигация в пространството на търсене.

Ограниченията на разгледаните подходи подчертават необходимостта от нови методи за индексиране, които могат ефективно да се справят с мащаба, размерността и сложността на съвременните информационни масиви, като същевременно осигуряват гъвкави възможности за филтриране. Хибриден подход за индексиране, предназначен да отговори на тези предизвикателства, е представен подробно в Глава 3 на настоящия труд.

2.5 Анализ на съвременните изследвания и научни пропуски

Представеният в тази глава литературен преглед очертава няколко ключови направления в областта на системите с интензивно използване на данни, архитектурите за машинно самообучение и обработката на високоизмерни данни. Настоящият раздел има за цел да консолидира основните изводи от този преглед и да идентифицира съществуващи празноти в научните познания.

2.5.1 Обобщение на съвременните изследвания

- **Системи с интензивно използване на данни:** Научните изследвания са установили основните характеристики на тези системи, подчертавайки техния фокус върху управлението и обработката на големи и сложни обеми от данни. Областта се развива с цел преодоляване на предизвикателствата, свързани с мащабируемостта, производителността и стратегиите за управление на данните [7], [8].

- **Машинно самообучение спрямо традиционни системи:** Съществува ясно разграничение между традиционните софтуерни системи и системите, базирани на машинно самообучение, като последните въвеждат специфични предизвикателства при проектирането, внедряването и оптимизацията [11], [15]. Адаптивният характер на системите с елементи на МС изисква нови подходи към практиките на софтуерното инженерство [16], [17].

- **Високоизмерни данни и търсене по сходство:** Постигнат е напредък в разработването на ефикасни алгоритми за търсене по сходство във високоразмерни пространства, като например ANN техниките (Approximate Nearest Neighbors) [32]. Въпреки това „проклятието на размерността“ остава постоянно предизвикателство [3].

- **Техники за индексиране и клъстеризация**: Предложени са различни подходи за управление и търсене в големи информационни масиви, включително графово-базирани методи [33] и обърнати индекси [35]. Решенията, базирани на графични процесори, показват значителен потенциал за търсене по сходство в милиардни мащаби [21].

- **Модели на данни и техники за представяне:** Разработени са техники за намаляване на размерността [23] и методи за обясним изкуствен интелект (Explainable AI, XAI) [36], [37] за справяне с предизвикателствата при интерпретацията на данните и прозрачността на моделите.

2.5.2 Идентифициране на научни пропуски

Литературният преглед разкрива няколко пропуска в съвременните изследвания в областта на системите с интензивно използване на данни, особено в контекста на приложенията за машинно самообучение. Една от ключовите области, изискващи по-нататъшно проучване, е интегрирането на сложни, многомерни възможности за филтриране с ефективни алгоритми за търсене по сходство при високоизмерни данни. В текущите изследвания липсват цялостни решения в тази насока, както подчертават Gollapudi и съавтори [31]. Този пропуск става все по-осезаем с нарастването на размерността и сложността на данните в съвременните приложения.

Идентифицираните научни пропуски подчертават необходимостта от непрекъснати иновации в системите с интензивно използване на данни, особено при тяхното взаимодействие с технологиите за машинно самообучение.

2.6 Обобщение

В настоящата глава беше направен преглед на съвременното състояние на научните изследвания в областта на системите с интензивно използване на данни, обхващащ широк спектър от теми – от фундаментални концепции до нови тенденции. Анализът на литературата разкрива еволюция в подходите на софтуерното инженерство, обусловена от предизвикателствата, свързани с управлението, обработката и анализа на големи и сложни обеми от данни в различни области.

Прегледът обхвана отличителните характеристики на тези системи, като изследва начините, по които те се различават от традиционните софтуерни системи по отношение на изискванията за мащабируемост, стратегиите за управление на данни и архитектурните съображения.

Ограниченията на съществуващите подходи, особено по отношение на ефективната обработка на сложни филтри при информационни масиви от порядъка на милиарди записи, мотивират разработването на нов хибриден метод за индексиране. В следващата глава подробно е описан предложеният метод, като се обръща внимание на неговото проектиране, прилагане и оценка в контекста на търсенето по сходство в милиарден мащаб.

Глава 3: Хибридно индексиране за търсене по сходство в милиарден мащаб

При анализа на архитектурните подходи в глава 1 и на изследванията в областта на индексирането и клъстеризацията в глава 2 бяха идентифицирани специфични ограничения на съществуващите методи за широкомащабни операции за търсене по сходство, особено по отношение на интегрирането на функционалности за многомерно филтриране. В настоящата глава е представен нов хибриден подход за индексиране, предназначен за справяне със сложността на операциите за търсене по сходство в милиарден мащаб, като същевременно осигурява разширени възможности за филтриране.

За преодоляване на тези предизвикателства са разработени различни видове алгоритми за приблизително търсене на най-близък съсед (Approximate Nearest Neighbors, ANN), като например IVF [33] и алгоритми, базирани на графи, като HNSW (Hierarchical Navigable Small World graphs) [33] графи. Тези методи целят постигането на баланс между точността на търсене и изчислителната ефективност, често чрез разделяне на пространството на търсене или конструиране на навигационни структури. Много от съществуващите подходи обаче изпитват затруднения при включването на сложни, многомерни критерии за филтриране, които са от съществено значение за прецизирането на резултатите от търсенето в практическите приложения.

В тази глава е представен алгоритъм, оптимизиран за работа с централен процесор (CPU) и опериращ с данни на диска, който интегрира плътни вектори с атрибути за дискретно филтриране чрез усъвършенствана структура от типа IVF-Flat. Предложеният метод използва техники за динамично управление на паметта за ефективна работа с информационни масиви, чийто обем надхвърля наличната оперативна памет. Това го прави особено подходящ за широкомащабно търсене по сходство при стандартни хардуерни конфигурации.

Както беше посочено в глава 2, широкомащабното търсене по сходство е свързано с множество предизвикателства, като там беше направен и преглед на съществуващи подходи за индексиране. Настоящата глава се фокусира върху интегрирането на многомерни филтри, обсъжда техники за оптимизация за централен процесор (CPU) и представя оценка на производителността на предложения хибриден метод. В заключение, дисертационният труд ще демонстрира практическата приложимост на подхода чрез анализ на конкретен казус, базиран на подмножество от информационния масив LAION-5B [38] – широкомащабна мултимодална колекция, съдържаща милиарди двойки „изображение-текст“.

Разработката, представена в тази глава, има за цел да предложи решение за справяне със сложността на търсенето по сходство в милиарден мащаб, предоставяйки усъвършенствани възможности за филтриране. Освен това тя се стреми да допринесе за създаването на по-ефективни и гъвкави системи за извличане на информация, способни да отговорят на изискванията на съвременните приложения с интензивно използване на данни.

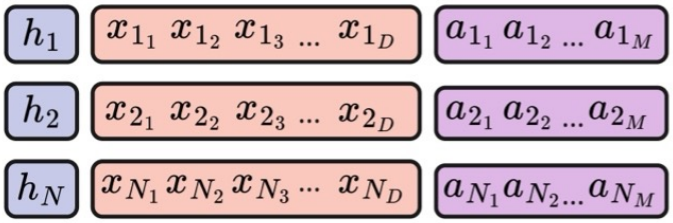
3.1 Предложен подход

В този раздел е представен предложеният подход за ефективно, широкомащабно търсене по сходство, което включва усъвършенствани възможности за многомерно филтриране.

3.1.1 Конструиране на хибридни вектори

Първата стъпка е да се конструират хибридните вектори. Хибридните вектори се означават като $H = \{h_1, h_2, \dots, h_N\}$, където всеки вектор $h_i \in \mathbb{R}^{(D+M)}$.

При наличие на набор от данни от N необработени високоразмерни вектори, обозначени като $\mathrm{X} = \{\mathrm{x}_1, \mathrm{x}_2, \dots, \mathrm{x}_N\}$, където всеки вектор $\mathrm{x_i} \in \mathbb{R}^{\mathrm{D}}$ и съответстващ набор от M атрибути за филтриране, обозначени като $\mathrm{A} = \{\mathrm{a}_1, \mathrm{a}_1, \dots, \mathrm{a}_M\}$, тогава се конструират хибридните вектори като $h_i = [x_i \,||\, a_i]$, където $\mathrm{h_i}$ е хибридният вектор, съответстващ на точката от данни i^{th}, $||$ означава конкатенация, N представлява общият брой вектори в набора от данни, а M представлява броят на наличните атрибути за филтриране (Изображение 4).



Изображение 4. Схема на конструиране на хибридните вектори

В Изображение 4 всеки ред съответства на точка от данни i^{th} и съответно представлява:

- h_1 до h_N – обозначение (индекс) на i-тия хибриден вектор, използвано във фигурата.
- x_{i_1} към x_{i_D} – компонентите на основното вграждане (първичния вектор x_i, с размерност D), което обикновено е генерирано от невронна мрежа.
- a_{i_1} до a_{i_M} – компонентите на атрибутния вектор a_i, съдържащ M стойности за филтриране.

Това унифицирано представяне на данните предлага няколко съществени предимства. То елиминира необходимостта от поддържане на множество отделни структури за индексиране, което от своя страна намалява разходите за съхранение и системна поддръжка. Същевременно, този подход осигурява гъвкаво и динамично филтриране.

Чрез комбинирането на плътни вектори за вграждане (dense embedding vectors) с дискретни атрибути за филтриране в единен хибриден вектор, се постига компактно представяне. Това представяне съдържа както информация за семантичното сходство, така и съответните метаданни за всяка точка от данните.

3.1.2. Изграждане на хибриден индекс

Изграждането на хибридния индекс включва няколко основни стъпки:

1) **Изчисляване на центроида:** Чрез прилагане на клъстеризиращ алгоритъм, като K-means[4] или MiniBatchKMeans[5], върху множеството от първични вектори $\mathrm{x_i} \in$

[4] KMeans:https://scikit-learn.org/stable/modules/generated/sklearn.cluster.KMeans.html
[5] MiniBatchKMeans: https://scikit-learn.org/stable/modules/generated/sklearn.cluster.MiniBatchKMeans.html

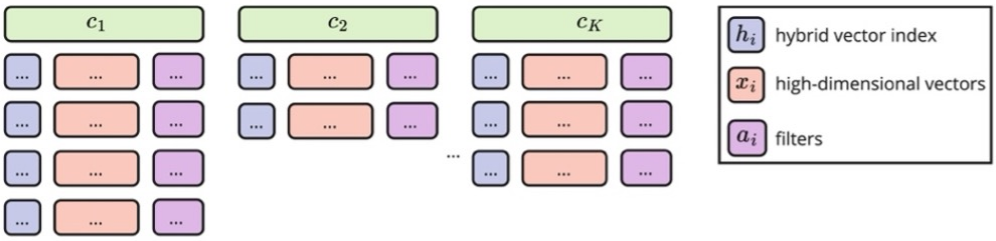
$\mathbb{R}^D$, за да се получат K клъстерни центроиди. Те се обозначават чрез $C = c_1, c_2, \ldots, c_K$, където всеки центроид $c_k \in \mathbb{R}^D$

2) **Присвояване на вектори към клъстери:** Всеки първичен вектор x_i се присвоява към най-близкия до него центроид c_k въз основа на избрана метрика за разстояние $d(\cdot,\cdot)$, (напр. евклидово разстояние или косинусово сходство, използвано като база за изчисляване на разстояние). По този начин се формират обърнати списъци (inverted lists), обозначени с $L = \{L_1, L_2 \ldots, L_K\}$, където всеки списък L_k съдържа индексите на първичните вектори, присвоени към центроида k^{th}.

3) **Съхранение на векторите (плоска структура):** За всеки обърнат списък L_k индексът съхранява пълните, некомпресирани представяния на първичните вектори. Този „плосък“ (flat) подход за съхранение означава, че се запазват пълните векторни данни, за разлика от указателите или компресираните, квантувани представяния.

4) **Асоцииране на атрибутите за филтриране:** В допълнение към основните вектори индексната структура поддържа и съответните атрибути на филтъра за всеки първичен вектор.

Получената индексна структура (изображение 5) позволява бързо идентифициране на релевантните клъстери по време на процеса на търсене. Освен това, тя дава възможност за ефективно прилагане на филтри и прецизно изчисляване на разстоянията в рамките на избраните клъстери.



Изображение 5. Илюстрация на структурата на хибридния индекс: центроиди, индексни елементи и асоциирани филтри

Броят на центроидите (K) в индекса IVF-Flat влияе върху компромиса между размера на индекса, времето за изграждане и ефективността на търсенето. Често срещана евристична практика е стойността на K да се избира в порядъка на $N/1000$ за информационни масиви (набори от данни) до 1 милион вектора или $\sqrt{N}$ за по-големи такива, където N е общият брой вектори[6].

3.1.3. Търсене

При зададен вектор на заявката $q \in \mathbb{R}^{(D+M)}$, съдържащ вход за търсене (основен вектор), обозначен като x_{input}, и списък с условия за филтриране като атрибутен вектор a_{input} , нашият метод извършва следните стъпки за извличане на топ-к най-сходни вектори, които отговарят на критериите за филтриране:

При зададен вектор на заявката $q \in \mathbb{R}^{(D+M)}$, който съдържа първичната част за търсене (първичен вектор), означена с $x_{input} \in \mathbb{R}^D$, и атрибутен вектор с условия за

[6] Pgvector, търсене на векторно сходство с отворен код за Postgres: https://github.com/pgvector/pgvector

филтриране $a_{input} \in \mathbb{R}^M$, предложеният метод извършва следните стъпки за извличане на k-те най-сходни вектора, които отговарят на зададените критерии:

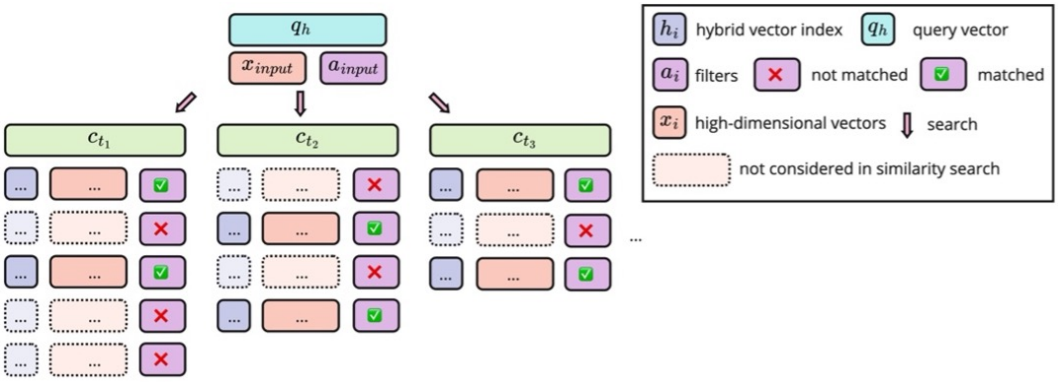
Стъпка 1: Конструиране на хибриден вектор на заявката q_h чрез конкатениране на първичния вектор на заявката x_{input} и представянето на условията за филтриране a_{input}, т.е. $q_h = [x_{input} \,||\, a_{input}]$.

Стъпка 2: Идентифициране на T най-близките центроиди до хибридния вектор на заявката q_h въз основа на метриката за разстояние на частта x_{input} . Тази стъпка стеснява пространството за търсене до най-обещаващите инвертирани списъци. Всички центроиди трябва да се съхраняват в паметта и всеки от целевите центроиди се обозначава с c_t, където $t = 1, 2, \ldots, T$ (Изображение 6)

Стъпка 3: Прилагане на условията за филтриране F (като атрибутен вектор) върху избраните обърнати списъци T чрез използване на структура в паметта за филтрите, като се отхвърлят всички вектори, които не удовлетворяват зададените ограничения a_{input}. Тази стъпка гарантира, че за последваща обработка се разглеждат само онези първични вектори, които отговарят на критериите за филтриране.

Стъпка 4: За всеки от векторите, преминали филтрирането в предходната стъпка, се изчислява точното разстояние между входния вектор на заявката x_{input} и съответния първичен вектор от обърнатия списък, като използвате оптимизирани процедури BLAS (Basic Linear Algebra Subprograms)[7] за ефективни операции с матрици.

Стъпка 5: Филтрираните и оценени резултати от обърнатите списъци, асоциирани с T-те центроида, се обединяват. От тях се избират k-те най-сходни вектора въз основа на изчислените им разстояния до x_{input}.



Изображение 6. Илюстрация на процедурата за търсене

Изборът на параметъра T (броят на най-близките центроиди, които се разглеждат в процеса на търсене) оказва значително влияние върху компромиса между точността на търсене и изчислителните разходи.

3.1.4. Избор на параметрите за клъстеризация и търсене

Производителността и точността на предложения хибриден подход за индексиране и търсене по сходство с филтриране зависят в голяма степен от два ключови параметъра: броя на центроидите (K), използвани при изграждането на индекса, и броя на най-

[7] BLAS (Basic Linear Algebra Subprograms), библиотека за операции с вектори и матрици: https://www.netlib.org/blas/

близките до заявката центроиди (T), които се обхождат по време на операциите по търсене.

Изграждане на индекс и избор на броя центроиди (K)

По време на етапа на изграждане на индекса, изборът на броя на центроидите K определя нивото на детайлност на разделянето на пространството от данни. По-голяма стойност на K води до по-фино сегментиране на векторното пространство, което потенциално подобрява точността на търсене, но това е за сметка на по-голям размер на индекса и по-дълго време за неговото изграждане.

Въз основа на емпирични наблюдения и в съответствие с утвърдените евристични методи в тази област задаването на K на приблизително $\sqrt{N}$, където N е общият брой вектори в набора от данни, осигурява ефективен баланс за набори от данни с милиарден мащаб.

Избор на параметъра за търсене T

Във фазата на изпълнение на заявката, параметърът T, представляващ броя на най-близките до заявката центроиди, които се обхождат, играе решаваща роля за постигане на баланс между точността и производителността на търсене. Малка стойност на T може да доведе до пропускане на релевантни резултати, особено при прилагане на сложни критерии за филтриране. От друга страна, голяма стойност на T може да доведе до по-бавно търсене поради по-големия брой векторни сравнения, които трябва да се извършат.

Изчислителната сложност на търсенето при предложения подход, макар теоретично да се доближава до $O(N)$ (при най-лош сценарий, например когато всички данни отговарят на критериите за филтриране), в практическите случаи обикновено позволява постигането на подлинейно време за изпълнение на заявките. Тази ефикасност се дължи на комбинацията от метода на центроидно базирано ограничаване на пространството на търсене (pruning) и разработения ефективен механизъм за филтриране. В глава 6 са представени емпирични резултати, които демонстрират ефективността на този подход при работа с информационни масиви от порядъка на милиарди записи.

В заключение, правилният избор на параметрите K и T, в съчетание с потенциални бъдещи подобрения като адаптивен избор на параметри и оптимизирани стратегии за дисково-базирано съхранение, утвърждава предложения хибриден подход за индексиране като надеждно и мащабируемо решение. Той е предназначен за търсене по сходство в милиарден мащаб и предлага разширени възможности за филтриране.

3.2 Ограничения

Изграждането на индекси за информационни масиви от порядъка на милиарди записи представлява значително изчислително предизвикателство. Потенциалните стратегии за смекчаване на този проблем включват прилагането на алгоритъма MiniBatchKMeans за ускорена клъстеризация или използването на предварително изградени индекси, ако са налични такива. От съществено значение е обаче да се отбележи, че тези подходи могат да доведат до намаляване на качеството на търсене, особено по отношение на пълнотата (recall), в сравнение с използването на стандартния K-means алгоритъм.

Освен това, стойностите на някои атрибути за филтриране може да изискват предварителна обработка, за да отговорят на специфични ограничения за съхранение

(например, формат от тип float32 [39]). Това от своя страна може да наложи извършването на процедури по нормализация или премащабиране.

Въпреки тези потенциални предизвикателства, предложеният алгоритъм представлява практично и икономически ефективно решение за търсене по сходство, предлагащо усъвършенствани възможности за филтриране.

3.3 Пример за търсене на сходство в милиарден мащаб

3.3.1 Експериментална уредба

За да се оценят ефективността и производителността на предложения хибриден алгоритъм за индексиране, беше проведен анализ на конкретен казус. За целта беше използван информационният масив LAION-5B – мащабна мултимодална колекция, която съдържа над 5 милиарда двойки „изображение-текст“. Проучването се фокусира върху подмножество от 1 милиард вграждания на изображения (Laion1B-nolang), всяко от които е представено като 768-измерен вектор, генериран от модела CLIP ViT-L/14 [40]. Това подмножество беше избрано с цел да се демонстрира способността на алгоритъма да извършва търсене по сходство във високоизмерни векторни пространства, прилагайки комплексни критерии за филтриране при работа с данни в голям мащаб.

Експерименталната постановка е проектирана за оценка на работата на предложения алгоритъм в условия, близки до реалните.

В Таблица 1 са описани основните параметри, използвани в рамките на този експеримент:

Таблица 1 . Параметри и стойности за оценка в милиарден мащаб

Параметър	Име	Стойност
N	Размер на набора от данни	1 милиард (10^9)
K	Брой центроиди ($\sim\sqrt{N}$)	32 000
D	Размерност на векторите	768
T	Брой на най-близките центрове за идентифициране	7
M	Брой атрибути за филтриране	10
V	Среден брой вектори на центроид	31 500

Предложеният алгоритъм е реализиран на програмния език Python с използването на библиотеката NumPy за ефективни числени изчисления. Всички експерименти са проведени на сървър със следните спецификации: централен процесор (CPU) Intel® Xeon® E-2274G @ 4,00GHz, оперативна памет (RAM) 64 GB DDR4, устройства за съхранение: 1 x 512 GB NVMe SSD + 2 x 6 TB SATA HDD.

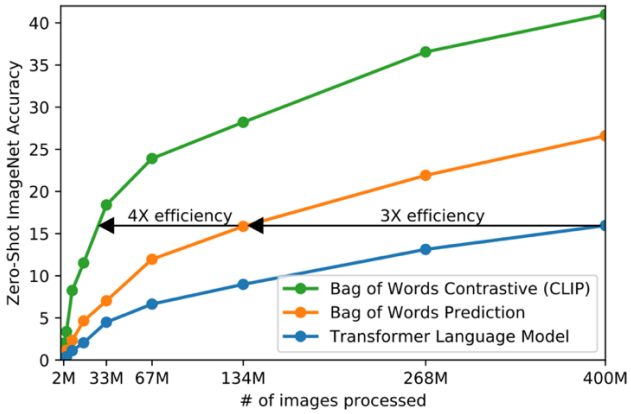
Конструкция на хибридните вектори

За целите на настоящия експеримент, хибридните вектори са конструирани чрез конкатенация на 768-измерните векторни вграждания, генерирани от модела CLIP, със синтетично генерирани атрибутни вектори. Към всяко CLIP вграждане е добавен 10-измерен атрибутен вектор (т.е. $M = 10$), предназначен да симулира метаданни. В резултат на това, общата размерност на получените хибридни вектори е 768+10=778.

Стойностите в атрибутните вектори са генерирани чрез равномерно разпределение в целочисления диапазон [-32768, 32767]; този диапазон е избран с оглед пълноценното

използване на типа данни float16 за постигане на ефективност на съхранението при запазване на необходимата точност.

Моделът CLIP е избран поради демонстрираната висока ефективност по отношение на точността му при класификация без предварителни примери (zero-shot accuracy) върху информационния масив ImageNet[8], както е представено на изображение 7.



Изображение 7. Точност на модела CLIP при класификация без предварителни примери (zero-shot accuracy) върху информационния масив ImageNet

Като алтернативен модел беше разгледан ImageBind [41], но при него дължината на генерираните векторни представяния е по-голяма, липсват публично достъпни, предварително изчислени информационни масиви от него, а лицензионните му условия са по-ограничителни за индустриална употреба.

Изграждане на индекс

Предвид изчислителната сложност, свързана с изграждането на индекси за информационни масиви от порядъка на милиарди записи, в рамките на настоящото проучване е използван съществуващ kNN индекс, предоставен с информационния масив LAION-5B. Приложени са допълнителни стъпки на обработка за интегриране на филтрите към този индекс. За да се оцени мащабируемостта на подхода за индексиране, са проведени и експерименти с алгоритъма MiniBatchKMeans от библиотеката scikit-learn, чрез който е изграден индекс за използваното подмножество от данни в рамките на няколко часа, използвайки сървър, оборудван само с централен процесор (CPU).

3.3.2 Оценка на производителността

Оценката на производителността на предложения алгоритъм за търсене по сходство в милиарден мащаб се фокусира върху ключови показатели като скорост и точност на търсене. Предложеният алгоритъм демонстрира подобрения по отношение на времето за търсене, като същевременно поддържа високи нива на точност.

В таблица 2 е представена подробна разбивка на времетраенето на отделните етапи от операцията за търсене:

[8] База данни с изображения, организирана в съответствие с йерархията на WordNet: https://www.image-net.org/

Таблица 2 . Производителност на търсенето

Операция	Време (секунди)
Търсене в центроиди	0.008
Филтриране	1.090
Подробно търсене в клъстери	0.330
Общо	1.428

Тези резултати подчертават високата ефективност на предложения алгоритъм, особено в етапа на идентификация на най-близките центроиди, който приключва за едва 8 милисекунди. Етапът на филтриране, макар и най-времеемкият (1.09 секунди), все пак представлява значително подобрение в сравнение с традиционните методи при работа с информационни масиви от такъв порядък.

Ограничения и перспективи за мащабируемост

Въпреки че предложеният алгоритъм демонстрира добри характеристики на мащабируемост, бяха идентифицирани някои ограничения:

1) Времето за изграждане на индекси за информационни масиви от порядъка на милиарди записи остава значително, макар това да е еднократен разход.

2) Едновременните (конкурентни) търсения могат да се превърнат в „тясно място“ (bottleneck) поради необходимостта от зареждане на различни части на индекса в оперативната памет.

3) В настоящата реализация са тествани само филтри, базирани на точно съвпадение на стойностите в атрибутните вектори. В бъдещи версии на алгоритъма би могло да се предвиди поддръжката на по-сложни релационни оператори за филтриране.

В заключение, проведената оценка на производителността и анализът на мащабируемостта показват, че предложеният алгоритъм предлага практично и икономически ефективно решение. Той е предназначен за търсене по сходство с усъвършенствани възможности за филтриране при работа с информационни масиви от порядъка на милиарди записи.

3.4 Обобщение

В настоящата глава беше представен нов хибриден подход за индексиране, предназначен да отговори на предизвикателствата на търсенето по сходство в милиарден мащаб, предлагащ усъвършенствани възможности за филтриране. Предложеният метод разширява класическата индексна структура от тип IVF-Flat чрез интегриране на плътни векторни вграждания с дискретни атрибути за филтриране.

Основните елементи на този подход включват: разработване на хибридно векторно представяне, което съчетава високоизмерни вграждания с атрибути за филтриране; използване на усъвършенствана структура от тип IVF-Flat, оптимизирана за обработка с централен процесор (CPU) и за дисково-базирано съхранение на данни; и осигуряване на ефективно интегриране на многомерни филтри в процеса на търсене.

В глава 6 са представени анализ на конкретен казус и подробни експериментални резултати, чрез които се оценяват производителността и ефективността на предложения метод в условия, близки до реалните. Възможните бъдещи изследователски направления включват разработването на допълнителни оптимизации и изследването на по-сложни сценарии за филтриране, с цел допълнително подобряване на възможностите и ефективността на системата.

Глава 4: Подобряване на интерпретацията на плътни вектори

В предходната глава беше представен хибриден подход за индексиране, предназначен за търсене по сходство в милиарден мащаб. Въпреки че постигането на висока производителност и мащабируемост е от съществено значение, интерпретацията на използваните в тези системи високоизмерни векторни представяния остава не по-малко значимо предизвикателство. То произтича както от високата им размерност, така и от специфичния им числов характер, които затрудняват интуитивното разбиране на съдържащата се в тях информация.

Съществуващите техники за интерпретация на вектори, включващи популярни методи за намаляване на размерността като PCA като PCA [22], t-SNE [23] и UMAP [24], както и различни подходи за визуализация, често не предоставят пряка и интуитивна интерпретация на отделните векторни представяния.

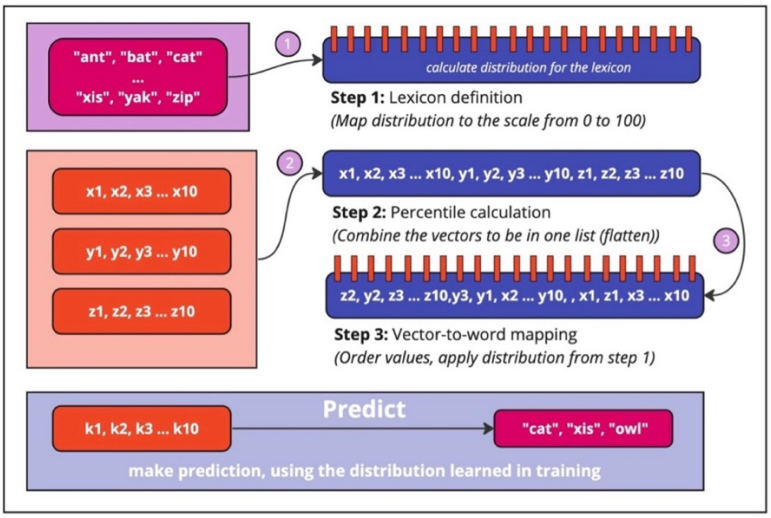
Трудностите при интерпретацията на високоизмерни данни се усложняват допълнително от т.нар. „проклятие на размерността“ (вж. раздел 2.3.1). Този феномен описва комплекс от явления, възникващи при анализ на данни във високоизмерни пространства, където с увеличаването на броя на измеренията обемът на пространството нараства експоненциално. В резултат на това наличните данни стават все по-разредени, което значително затруднява откриването на надеждни закономерности или зависимости.

В следващите секции на тази глава ще бъдат разгледани съществуващи подходи за справяне с това предизвикателство. Ще бъде представен и LangVec [6] – нов метод, предназначен за подобряване на интерпретируемостта на плътните векторни представяния чрез тяхното съпоставяне с лесно четими за човека лексикални репрезентации.

4.1 Предложен подход

Същността на предложения подход, който е реализиран в рамките на метода LangVec, се състои в съпоставянето на високоизмерни числови вектори с лексикални репрезентации, които са лесно четими за човека. Това се постига чрез механизъм за съпоставяне, базиран на персентилно разпределение на стойностите от векторите спрямо думи от предварително дефиниран лексикон.

Процесът може да бъде разделен на три основни стъпки: (1) дефиниране на лексикона; (2) изчисляване на персентилите; и (3) съпоставяне на векторни стойности с думи от лексикона. Тези стъпки са илюстрирани на изображение 8 и са описани подробно в следващите подраздели.



Изображение 8. Илюстрация на работния процес на LangVec, от входни вектори до лексикални представяния

4.1.1 Дефиниране на лексикона

Първата стъпка в рамките на предложения подход е дефинирането на лексикон – набор от думи или фрази, които ще служат за лексикално представяне на векторните вграждания. По подразбиране, методът LangVec използва предварително дефиниран лексикон от често срещани, кратки английски думи, но е предвидена и възможност потребителите да предоставят собствени, специфични за нуждите си лексикони. Изборът на лексикон позволява интерпретация, съобразена със специфичната предметна област, и адаптиране към конкретните данни и сценарии на приложение.

Разпределението на лексикона D се изчислява по следния начин:

$$D = (d_1, d_2, \dots, d_{L-1}) \quad (1)$$

където L е размерът на лексикона, а $d_i = i * \frac{100}{L-1}$ за $i = 1, 2, \dots, L-1$

Това показва как персентилите $L - 1$ за съпоставяне на величини на вектори с думи са равномерно разпределени в диапазона $[0, 100]$. Разпределението на лексикона се изчислява, като се използват равномерно разположени персентили, за да се осигури балансирано съотнасяне на векторните величини към думите в целия диапазон $[0, 100]$.

4.1.2 Изчисляване на персентилите

За осъществяване на съпоставянето между векторните стойности и думите от лексикона, е необходимо първо да се изчислят конкретните прагови стойности (персентили). Това става въз основа на емпиричното разпределение на стойностите в наличните вектори от обучаващия набор данни. Функцията fit приема списък с числови вектори и изчислява персентилите, съответстващи на всяка дума в лексикона.

Изчисляване на персентил: При наличие на набор от N числови вектори $V = \{v_1, v_2, \dots, v_N\}$, където всеки вектор v_j има M измерения, персентилите P се изчисляват по следния начин:

$$P = (p_1, p_2, \dots, p_{L-1}) \quad (2)$$

Където $p_i = percentile(flatten(V),\ d_i)$ за $i = 1, 2, \dots, L-1$

Функцията $flatten(V)$ приема като входен аргумент набора от N числови вектори V и конкатенира (сплесква) всички техни елементи в един общ едномерен масив. Тази операция опростява процеса на изчисляване на персентилите, разглеждайки всички индивидуални стойности от векторите като единна съвкупност от данни. Например, ако V се състои от три вектора, $[0.1, 0.2, 0.3]$, $[0.4, 0.5, 0.6]$ и $[0.7, 0.8, 0.9]$, то $flatten(V)$ ще върне масива $[0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9]$.

Функцията $percentile(X, d)$ изчислява персентила d^{th} на данните в едномерния масив X. Персентилът d^{th} е стойността, под която попадат d процента от данните. Например, ако $d = 50$, функцията ще върне медианната стойност на данните в X. Ако $d = 25$, тя ще върне първия квартил ($Q1$), а ако $d = 75$, ще върне третия квартил ($Q3$). Функцията за персентил се използва за определяне на праговите стойности за съпоставяне на величините на векторите с думите в лексикона.

4.1.3 Съпоставяне на векторни компоненти с думи от лексикона

След като бъдат изчислени персентилните прагове, методът LangVec позволява съпоставянето на нови числови вектори със съответните им лексикални репрезентации. Това се осъществява чрез функцията predict (предсказване). Тя приема като входен аргумент един числов вектор и за всяка негова част (chunk) – чийто размер по подразбиране е 3 последователни компоненти – определя съответстваща дума от лексикона. Принадлежността към дадена дума се установява въз основа на сравнението на средната аритметична стойност на компонентите в анализираната част с предварително изчислените персентилни прагове

При наличие на входен вектор v с размери M, лексикон W с размер L и набор от проценти $P = (p_1, p_2, \ldots, p_{L-1})$, LangVec съпоставя v с поредица от думи $S = (s_1, s_2, \ldots, s_K)$, като използва следните стъпки:

1) Входният вектор v се разделя на K части, всяка от които съдържа C елементи, където C е размерът на частта. Броят на частите K се определя от:

$$K = \lfloor M / C \rfloor \text{ (разпределение по групите)} \quad (3)$$

Този подход гарантира, че се обработват само пълни части от вектора с размер C. Елементите в края на вектора v, ако техният брой е по-малък от C, не се използват за формиране на последна дума в поредицата S.

2) За всяка част c_j, където $j = 1, 2, \ldots, K$, се изчислява нейната средна аритметична стойност m_j:

$$m_j = mean(c_j) \quad (4)$$

3) Всяка изчислена средна стойност m_j се сравнява с персентилните прагове от набора P, за да се определи съответната дума s_j от лексикона W.

$$s_j = W[b_j] \quad (5)$$

където $b_j = sum(m_j > p_i \text{ for } i = 1, 2, \ldots, L - 1)$, съответно b_j представлява броя на процентите, които m_j надвишава, а $W[b_j]$ извлича думата с индекс b_j в лексикона W.

4.2 LangVec: Оценка на лексикалното представяне

Подходът LangVec, предназначен за подобряване на интерпретацията на високоизмерни числови вектори, беше подложен на обстойна оценка с цел анализ на неговата производителност, ефикасност и ефективност (качество на интерпретацията) при сценарии, близки до реалните. В този раздел е представена методологията на проведената оценка, включваща сравнителен анализ, получените резултати за производителност и анализ на постигнатата интерпретируемост.

4.2.1 Методология за сравнителен анализ

За оценката на библиотеката LangVec беше приложена многоаспектна методология за сравнителен анализ (бенчмаркинг), базирана на използването както на синтетични, така и на реални информационни масиви (набори от данни).

Експериментална постановка

Сравнителният анализ (бенчмаркинг) беше проведен на сървър със следните спецификации: централен процесор (CPU) Intel® Xeon® E-2274G @ 4,00GHz; оперативна памет (RAM) 64 GB DDR4; устройство за съхранение (SSD) 512 GB NVMe.

Тази хардуерна конфигурация е идентична с използваната при анализа на казуса за търсене по сходство в милиарден мащаб (описан в глава 3), което дава възможност за директни сравнения на производителността и отразява типични практически среди за високопроизводителни изчисления.

Подготовка на набора от данни

В процеса на сравнителен анализ бяха използвани набори от данни с разнообразна размерност, за да се оцени работата на LangVec в различни мащаби:

1) **Синтетични набори от данни:** Бяха генерирани случайни вектори с размери от 256 до 1024, а размерите на наборите от данни варираха от 10^3 до 10^6 вектора.
2) **Набор от данни от реалния свят:** База данни PostgreSQL[9] беше попълнена с 10 милиона низове, всеки с дължина 32 символа, съставени от малки букви от „a” до „z”.

Показатели за оценка

При оценката на резултатите на LangVec беше използван изчерпателен набор от показатели. Те включват време за обучение на модела (fit time), което измерва способността му да се адаптира към информационни масиви (набори от данни) с различен обем, и време за генериране на лексикални представяния (predict time), което измерва ефективността (бързодействието) на този процес.

Процедура за сравнителен анализ:

1) **Оценка върху синтетични информационни масиви::**
   a) Генерирани са набори от данни с размери 10^3, 10^4, 10^5 и 10^6 вектори, всеки с 256 измерения.

[9] База данни PostgreSQL: https://www.postgresql.org/

b) Измерено е времето за обучение (fit time) на модела LangVec за всеки от генерираните масиви.

c) Извършени са по 10 000 операции за генериране на лексикални представяния (прогнози) с използване на всеки обучен модел, като е записано средното време за една прогноза (predict time).

2) Оценка върху информационен масив от реално приложение (сравнителен сценарий):

a) Имплементирана е функция **find_similar_strings** в СУБД PostgreSQL за извършване на търсене на сходство на низове с помощта на разстоянието на Левенщайн.

b) Изпълнени заявки с различни максимални разстояния на Левенщайн (5, 10 и 20).

c) Измерено време за изпълнение на всяка заявка.

d) Проведен е и допълнителен експеримент за оценка на способността на LangVec да генерира последователно идентични лексикални представяния за идентични входни вектори (тест за точност на съответствието с 10 произволни вектора).

Тази методология за сравнителен анализ е разработена с цел да осигури задълбочено разбиране на аспектите на производителност, мащабируемост и практическа приложимост на LangVec както при малки експериментални постановки, така и при работа с големи информационни масиви от реални приложения.

4.2.2 Резултати за производителността

Проведеният сравнителен анализ (бенчмаркинг) предостави данни за производителността на LangVec при работа с набори от данни с различен обем и характеристики. Резултатите демонстрират ефективността и мащабируемостта на разработената библиотека при осъществяване на съпоставянето между високоизмерни вектори и техните лексикални представяния.

Изпълнение на синтетичен набор от данни

Таблица 3 са представени показателите за ефективност на LangVec върху синтетични набори от данни с различни размери:

Таблица 3 . Сравнителни резултати, показващи производителността на LangVec при различни размери на наборите от данни

Брой вектори	Време за обучение (секунди)	Време за прогнозиране (секунди)
10^3	0.0130	$4.25 * 10^{-4}$
10^4	0.0925	$4.11 * 10^{-4}$
10^5	0.8658	$4.53 * 10^{-4}$
10^6	9.9734	$4.13 * 10^{-4}$

Основни наблюдения:

1) **Време за обучение (fit time):** Времето за обучение на модела нараства линейно с обема на обучаващия набор данни, което е индикация за добра мащабируемост.

Например, за информационен масив от един милион вектора, процесът на обучение (fit) приключва за по-малко от 10 секунди.

2) **Време за генериране на представяне (predict time, inference):** Времето за генериране на лексикално представяне (прогнозиране) остава постоянно ниско (около 0,4 ms.) независимо от обема на обработваните данни. Това е показател за висока ефективност при преобразуването на векторни данни в лексикални представяния.

Запазване на локалността (locality preservation)

Методът LangVec за съпоставяне, базиран на персентилно разпределение, демонстрира добри свойства по отношение на запазването на локалността. Малки изменения в стойностите на компонентите на входните вектори водят до минимални или само локализирани промени в генерираното лексикално представяне. Това свойство улеснява интерпретацията на семантичните връзки (сходства и разлики) между отделните точки от данните.

Сравнителни предимства

В сравнение с традиционните техники за намаляване на размерността, като PCA или t-SNE, LangVec предлага специфичното предимство да генерира резултати, които са директно четими и интуитивно разбираеми за човека.

В заключение, представените резултати от оценката на производителността и интерпретируемостта показват, че LangVec представлява ефективно и гъвкаво решение за съпоставяне на високоизмерни вектори с лесно интерпретируеми лексикални представяния.

4.3 Обобщение

В настоящата глава беше представен LangVec – метод за подобряване на интерпретируемостта на високоизмерни числови векторни представяния. Основно внимание в главата беше отделено на методологията за трансформиране на плътни векторни представяния в лесно четими за човека лексикални репрезентации, която използва подход за съпоставяне, базиран на персентилно разпределение.

В рамките на главата бяха обсъдени също така практическото приложение на LangVec, включително използването му в различни сценарии, и неговата чувствителност към изменения във входните данни. Бяха изтъкнати предимствата на LangVec по отношение на подобрената интерпретируемост, гъвкавостта и производителността, особено в приложения като семантично търсене, откриване и премахване на дубликати на данни, и клъстеризация.

В следващата глава (глава 5) ще бъдат разгледани моделите на данни за системи с интензивно използване на данни, като специално внимание ще бъде обърнато на колонно-ориентираните подходи.

Глава 5: Модели на данни за системи с интензивно използване на данни

С непрекъснатото развитие на системите с интензивно използване на данни, необходимостта от ефективни механизми за съхранение и извличане на информация придобива все по-голямо значение. Традиционните редово-ориентирани бази данни, макар и подходящи за множество приложения, често се сблъскват с предизвикателства по отношение на производителността при работа с мащабни натоварвания от аналитичен тип. В настоящата глава ще бъдат разгледани алтернативни модели на данни, с акцент върху колонно-ориентираните архитектури, които предлагат решения за тези предизвикателства.

5.1 Проектиране на модел, ориентиран към колони

Ключов компонент на предложения колонно-ориентиран модел е алгоритъмът, използван за генериране на оптимална колонна структура. Този алгоритъм се адаптира динамично към специфичните изисквания на системи, базирани на уеб известия (webhooks) – механизми за автоматизирано уведомяване в реално време между различни софтуерни приложения, което се задейства при настъпване на определени събития. По този начин се осигурява оптимизиран модел на данните, съобразен с характеристиките на потока от данни, генериран от тези известия. Представеният по-долу псевдокод илюстрира процеса на вземане на решения за всяко поле от данните на тези уеб известия::

```
WHILE new field EXISTS
   CASEWHERE field purpose
       WHEN repetitive event with multiple choices:
           IF event column EXISTS AND current choice NOT IN list THEN
               ADD current choice TO list
           ELSE
               CREATE column WITH enumeration
           ENDIF
       WHEN unique identifier for non-main entity:
           SKIP field
       OTHERWISE:
           CREATE column WITH appropriate field type
   ENDCASE
ENDWHILE
```

Листинг 1 . Псевдокод на алгоритъма за генериране на предложения колонно-ориентиран модел на данни

Този алгоритъм обработва итеративно всяко поле от данните, като взема решение за начина на неговото представяне в колонно-ориентираната структура. Алгоритъмът пропуска уникалните идентификатори, отнасящи се до второстепенни (неосновни) обекти, с цел поддържане на опростена структура на модела. За всички останали полета се създават колони със съответните типове данни, осигурявайки изчерпателно представяне на информацията. В резултат на този подход се получава адаптивен колонно-ориентиран модел, който постига баланс между гъвкавост, производителност и ефективност на съхранението, съобразен със специфичните изисквания на системата за обработка на уеб известия (webhooks).

Тази денормализация на данните е целенасочена и служи за оптимизиране на производителността в контекста на колонно-ориентираното съхранение и обработката на интензивни потоци от данни. Моделът е структуриран около една основна таблица,

наречена webhooks. Тази таблица съдържа цялата необходима информация за всяко събитие, свързано с уеб известие (webhook), включително метаданни, детайли по изпратената заявка и информация за получения отговор.

За ефективното прилагане на този модел в колонно-ориентирана база данни е предложена денормализирана структура на данните. Този подход се отклонява от традиционните нормални форми, използвани в редово-ориентираните релационни бази данни, като основната цел е повишаване на производителността в средите с интензивно използване на данни, особено при често изпълнявани аналитични операции.

5.2 Модел на данни, ориентиран към колони, за системи, базирани на уеб известия (webhooks)

В този раздел на дисертацията е представен анализ на колонно-ориентиран модел на данни, проектиран за системи, базирани на уеб известия (webhooks). Акцентът е поставен върху неговото прилагане и характеристиките на производителност в среди с интензивно използване на данни.

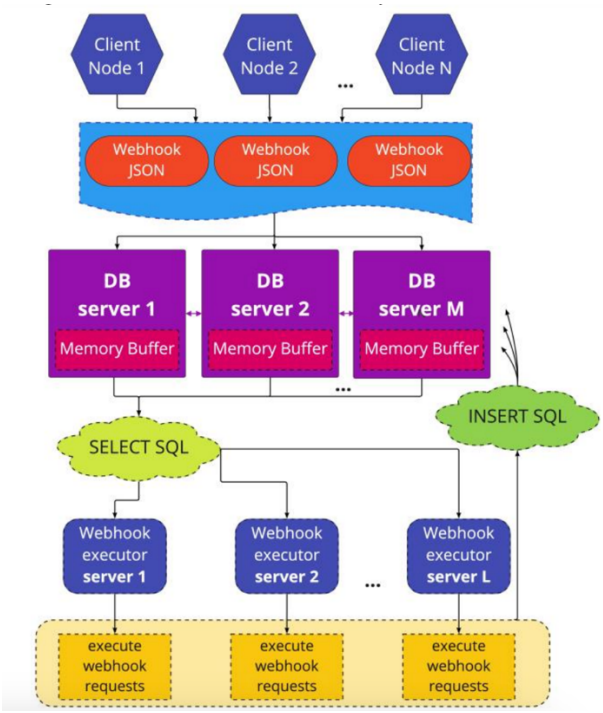
5.2.1 Проектиране и внедряване на системата

Архитектурата на системата за уеб известия (webhooks) включва колонно-ориентиран модел на данни с цел улесняване на ефективното съхранение, извличане и обработка на данни за събития при работа в голям мащаб. Системата се състои от следните ключови компоненти:

1) **Разпределени сървъри за бази данни:** Множество разпределени сървъри съхраняват данните от уеб известията (webhooks), използвайки колонно-ориентиран подход. Това допринася за отказоустойчивостта и мащабируемостта на системата.

2) **Сървъри за обработка на уеб известия (webhook executors):** Тези сървъри извличат нови задачи (събития), изпълняват съответните HTTP заявки към целеви системи и съхраняват получените резултати обратно в базата данни.

3) **Клиентски възел (client node):** Специализиран сървър, отговорен за приемането (ingestion) на конфигурациите на уеб известията и данните от тях, както и за последващото им зареждане в сървърите на базата данни.



Изображение 9. Архитектура на колонно-ориентираната система за уеб известия

Изображение 9 е представена архитектура на реализираната колонно-ориентирана система за уеб известия (webhooks). Илюстрирано е взаимодействието между клиентските възли, сървърите на базата данни (които използват буфери в оперативната памет) и сървърите за обработка на уеб известия. Показан е и потокът от данни, осъществяван чрез SQL операциите SELECT и INSERT.

При конкретната реализация на системата е използвана ClickHouse – колонно-ориентирана система за управление на бази данни (СУБД). За създаване на основната таблица за съхранение на данните от уеб известията е използвана следната SQL заявка:

```
CREATE TABLE webhooks
(
    id UUID,
    created_at DateTime,
    updated_at DateTime,
    status Enum8('added' = 1, 'processed' = 2, 'failed' = 3),
    endpoint String,
    http_method Enum8('GET' = 1, 'POST' = 2, 'PUT' = 3, 'DELETE' = 4),
    payload String,
    response_code UInt16,
    response_body String
) ENGINE = MergeTree()
ORDER BY (created_at, id);
```

Листинг 2. SQL заявка за дефиниране на таблицата webhooks в системата за уеб известия

Така дефинираната схема на таблицата улеснява извличането на данни и тяхната аналитична обработка, особено при заявки, филтрирани по времеви критерии, и при филтриране по статус на събитията.

5.2.2 Сравнителен анализ на производителността

За да се оцени производителността на колонно-ориентирания подход, беше извършен сравнителен анализ с традиционна редово-ориентирана система за управление

на бази данни (СУБД). В експеримента са използвани PostgreSQL 12.9 като редово-ориентирана СУБД и ClickHouse 22.2.2.1 като колонно-ориентирана СУБД. Двете системи са тествани на идентичен хардуер: виртуален сървър с 4 GB оперативна памет (RAM), 2 виртуални централни процесора (vCPU), 80 GB SSD устройство за съхранение, работещ под операционна система Ubuntu 20.04 в облачната платформа AWS Lightsail[10].

Анализът на производителността се фокусира върху седем тестови сценария, като всеки от тях оперира върху информационен масив (набор от данни), съдържащ 1 000 000 записа:

1. Четири групови операции SELECT върху записи със статус: (**added**, **tried_1**, **failed**, **finished**).
2. Две операции SELECT с ограничение на броя върнати редове (с клауза LIMIT) върху групата записи със статус **added**.
3. Една аналитична заявка, агрегираща (обобщаваща) кодовете на HTTP отговорите.

В Таблица 4 са представени сравнителните резултати от този анализ:

Таблица 4. Резултати от сравнителния анализ на производителността на колонно-ориентирания подход

Тип заявка	Редове (секунди)	Колони (секунди)	Подобрение в производителността
Group (added)	1.73	1.03	40.46%
Group (tried_1)	1.20	0.77	35.83%
Group (failed)	0.47	0.44	6.38%
Group (finished)	2.21	1.30	41.18%
Group (added) with LIMIT (5000)	0.55	0.35	36.36%
Group (added) with LIMIT (20000)	0.60	0.37	38.33%
Response code analysis	0.14	0.02	85.71%

Основните констатации от анализа на производителността включват:

1) **Подобрена производителност на заявките:** Колонно-ориентираният подход демонстрира по-висока производителност при всички типове тествани заявки, като подобренията варират от 6,38% до 85,71%.

2) **Мащабируемост спрямо обема на данните:** При заявки, връщащи по-голям обем резултати (например, заявката за групата записи със статус **finished**), колонно-ориентираният модел показва подобрение от 41,18%, което е индикация за по-добра мащабируемост с увеличаване на обема на данните**.**

3) **Ефективност при аналитични заявки:** Аналитичната заявка за агрегиране на кодовете на HTTP отговорите демонстрира най-значителното подобрение (85,71%), което подчертава предимствата на колонно-ориентирания модел при аналитична обработка на данни.

4) **Подобрение при филтрирани заявки:** Заявките, използващи клаузата LIMIT, показват устойчиво повишаване на производителността (между 36% и 38%),

[10] Amazon Lightsail: https://aws.amazon.com/lightsail/

което демонстрира ефективността на колонно-ориентирания подход в сценарии, изискващи селективно извличане на данни.

Резултатите показват, че колонно-ориентираният модел на данни предлага значителни предимства по отношение на производителността на системи за уеб известия (webhooks), особено при сценарии, включващи мащабна обработка на данни и аналитични заявки. Подходът демонстрира висока ефикасност при обработката на данни от тип времеви редове и при филтриране по статус на събитията – операции, които са често срещани изисквания в архитектури, базирани на уеб известия (webhooks) и управлявани от събития (event-driven).

В заключение, моделът на данни, ориентиран към колони, представлява обещаващ подход за оптимизиране на системи за уеб известия (webhooks) и други подобни приложения, характеризиращи се с интензивна обработка на данни. Способността му да намалява времето за изпълнение на заявките, особено при аналитични натоварвания, утвърждава неговата приложимост за системи, които обработват големи обеми от данни за събития и изискват чести аналитични операции.

5.3 Обобщение

Направените в настоящата глава (глава 5) констатации показват, че колонно-ориентираните модели на данни, когато са правилно проектирани, реализирани и оптимизирани, предлагат съществени предимства при тяхното приложение в системи с интензивно използване на данни. Тези предимства включват подобрена производителност при изпълнение на заявки, по-ефективно компресиране на данните и повишена мащабируемост. Подобни характеристики правят колонно-ориентираните подходи особено подходящи за съвременните приложения с интензивно използване на данни, които изискват ефективна обработка на мащабни аналитични натоварвания.

Приложимостта на колонно-ориентираните модели на данни в широк спектър от сценарии се дължи на техните фундаментални характеристики. Колонно-ориентираният им формат за съхранение на данни, съчетан с усъвършенствани техники за компресиране и оптимизирани механизми за обработка на заявки, позволява ефективна обработка на високоизмерни данни, извършване на сложни изчисления и справяне с мащабни аналитични натоварвания.

Понеже обемът и сложността на данните в съвременните приложения продължават да нарастват (изображение 1), гъвкавостта и предимствата на колонно-ориентираните модели на данни ги утвърждават като ключов компонент в архитектурата на софтуерните системи с интензивно използване на данни.

Глава 6: Експериментални изследвания

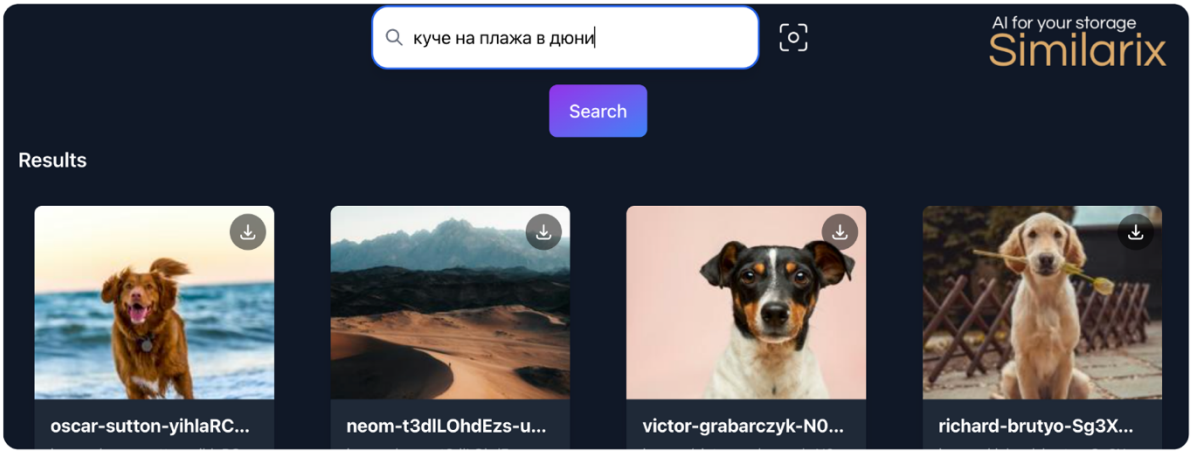
В настоящата глава са представени поредица от експериментални изследвания и сравнителни анализи, предназначени за оценка на ефективността (постигане на целите), ефикасността (оптимално използване на ресурси) и мащабируемостта на предложените в дисертационния труд методи за реализация на софтуерни системи с интензивно използване на данни. Представените тук експерименти служат за допълнително валидиране и цялостна оценка на подходите, разработени в Глава 3, Глава 4 и Глава 5.

6.1 Общ преглед

Експерименталните изследвания в тази глава са организирани с цел поетапна оценка на ефективността и практическата приложимост на предложените подходи. Те започват с оценка на решенията за мащабно търсене по сходство и преминават през анализ на различни техники за представяне и обработка на данни, както и тяхното интегриране. Тази систематична оценка цели да осигури изчерпателно разбиране за това как предложените методи функционират в реални сценарии и какви са техните предимства спрямо съществуващи решения.

6.2 Интегриране в производствена система

Теоретичните подходи и експерименталните резултати, разгледани в предходните глави (глави 3, 4 и 5), бяха успешно интегрирани в реална производствена система. Това демонстрира тяхната практическа приложимост и ефективност в реални условия и послужи за валидиране на концепцията (proof-of-concept). В настоящия раздел се описват архитектурата на тази система и детайлите по нейното производствено внедряване.

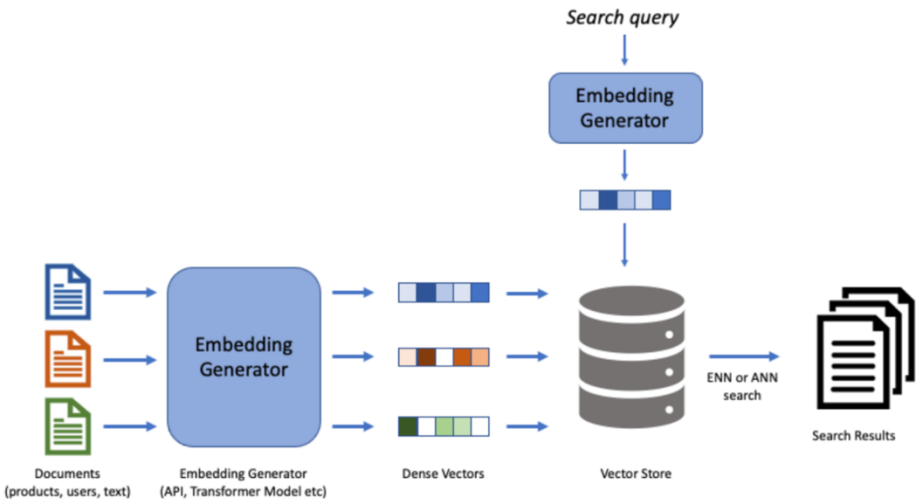


Изображение 10. Визуален преглед на производствената система за семантично търсене

6.2.1 Архитектура на системата

Описаната производствена система е предназначена за изпълнение на широкомащабни задачи, свързани със семантично търсене и извличане на данни по сходство. Тя интегрира разработения хибриден подход за индексиране на информационни масиви от порядъка на милиарди записи (представен в Глава 3), метода LangVec за подобряване на интерпретацията на векторни представяния (описан в Глава 4) и колонно-ориентирания модел на данни (разгледан в Глава 5), използван за целите на последващ анализ.

Изображение 11 илюстрира архитектурата на високо ниво на тази система.



Изображение 11. Архитектура на високо ниво на производствената система за семантично търсене

Архитектурата на системата се състои от няколко основни компонента:

1) **Приемане на документи (document ingestion):** Системата е проектирана да приема разнообразни типове входни документи (напр. описания на продукти, потребителски данни, текстови корпуси).

2) **Генератор на векторни вграждания (embedding generator):** Този компонент, реализиран с помощта на съвременни модели на трансформатори (като CLIP), преобразува входните документи в плътни векторни представяния.

3) **Хранилище за вектори (vector store):** Този ключов компонент отговаря за съхранението и индексирането на генерираните плътни векторни представяния. Конкретната реализация на хранилището може да варира (напр. да бъде раздвоена) в зависимост от мащаба на обработваните данни.

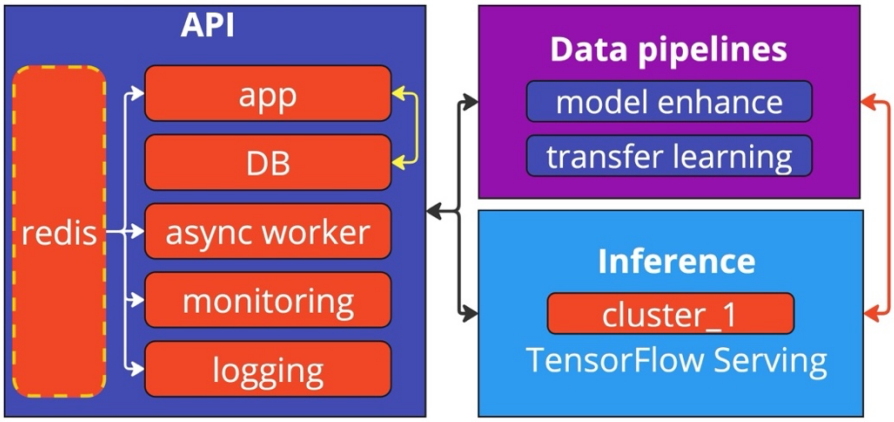
4) **Обработка на заявки за търсене (search query processing):** Входящите потребителски заявки за търсене преминават през аналогичен процес на генериране на векторно вграждане, преобразувайки се във вектори от същото пространство, в което са представени и индексираните документи.

5) **Извършване на търсене (ENN/ANN search):** Системата извършва търсене по точност (exact nearest neighbor, ENN) или по приблизителност (Approximate Nearest Neighbor, ANN) в зависимост от избрания метод за индексиране и специфичните изисквания на заявката.

6) **Извличане и представяне на резултати (result retrieval and presentation):** Резултатите от търсенето се извличат и представят на потребителя, като е възможно прилагането на допълнителни стъпки за последваща обработка или ранжиране (класиране) на резултатите.

Производствената система е изградена с помощта на съвременен набор от технологии (технологичен стек) с цел осигуряване на висока производителност, мащабируемост и лесна поддръжка:

- **Backend рамка (framework):** Django[11], избрана заради своята надеждност и обширна екосистема.
- **Асинхронна обработка на задачи:** Celery[12], използвана за ефективна обработка на пакетни и фонови задачи.
- **Frontend:** Tailwind CSS[13] за реализация на отзивчив и лесно адаптивен потребителски интерфейс.
- **Векторна обработка:** Специално разработени модули, използващи библиотеката Vector Forge[14] за ефективни векторни манипулации. Vector Forge е библиотека на програмния език Python, разработена за улесняване на преобразуването на различни типове данни във векторни представяния (характерни вектори). Тя поддържа множество предварително обучени модели за векторизиране на изображения и текст, включително CLIP ViT-B/32, CLIP ViT-L/14, VGG16, VGG19 и Xception, като по този начин предлага гъвкавост при обработката на различни типове данни и размерности.



Изображение 12 . Архитектура на по-ниско ниво на производствената система за семантично търсене

Тази диаграма илюстрира детайлизираната архитектура на описаната производствена система, като акцентира върху три основни компонента: програмен интерфейс (API), конвейери за обработка на данни (data pipelines) и модули за логически извод (inference engines). API компонентът управлява основните функционалности, включително бизнес логиката на приложението, взаимодействията с базата данни, асинхронната обработка на задачи, мониторинга и регистрирането на събития (logging), като координацията между тези елементи се осъществява чрез брокер на съобщения Redis.

Архитектурата на тази производствена система демонстрира практическото приложение на резултатите от проведените изследвания. Тя показва как теоретичните постижения в областта на търсенето по сходство и интерпретацията на векторни представяния могат да бъдат ефективно използвани в реални, широкомащабни приложения.

[11] Уеб рамка Django: https://www.djangoproject.com/
[12] Опашка за задачи на Celery: https://docs.celeryq.dev/en/stable/getting-started/introduction.html
[13] TailwindCSS: https://tailwindcss.com/
[14] Библиотека Vector Forge: https://github.com/s-emanuilov/vector_forge

6.2.2 Показатели за производителност в реални сценарии

Производителността на описаната производствена система е оценена в различни реални сценарии с цел оценка на нейната ефективност (постигане на качествени резултати) и ефикасност (бързодействие и използване на ресурси). Оценките са фокусирани върху ключови показатели, които отразяват както техническите ѝ характеристики, така и потенциалното ѝ въздействие върху потребителското изживяване в практическите приложения.

При работа с информационни масиви, съдържащи до 30 милиона вектора, и използване на pgvector с индексиране от тип HNSW, системата постига медианно време за отговор от около 500 ms (милисекунди). При преминаване към по-големи информационни масиви (над 30 милиона вектора) и използване на предложения хибриден подход за индексиране в милиарден мащаб, системата демонстрира очаквано по-високи, но все още приемливи времена за отговор, с медианно време за изпълнение на заявка от 1,428 секунди.

Тези резултати, макар и не толкова високоскоростни, колкото при някои комерсиални системи, оптимизирани за работа с по-малки информационни масиви, представляват значително постижение в контекста на търсенето по сходство в милиарден мащаб. Възможността да се поддържа време за отговор под 1,5 секунди при обработка на информационни масиви от такъв порядък, особено при работа върху изчислителна инфраструктура, базирана единствено на централни процесори (CPU), демонстрира високата ефикасност на предложения хибриден подход за индексиране.

Пълнотата (recall), ключов показател при оценката на алгоритми за приблизително търсене на най-близък съсед (ANN), показва добри резултати при информационни масиви с различен обем. Системата постоянно постига над 90% recall@10 както за информационни масиви с умерен обем, така и за такива с обем от порядъка на милиарди записи.

6.2.3 Сравнителен анализ и съпоставка на моделите

В рамките на настоящото изследване е направен и научен принос чрез публикуването на отворен набор от данни (open dataset)[15]. Този информационен масив, описан подробно в Допълнение В, се състои от обработени изображения от набора данни COCO, като за целта е използван моделът CLIP ViT-L/14. Публикуваният набор предоставя 768-измерни векторни представяния за приблизително 123 000 изображения и представлява ценен ресурс за изследователи и практици, работещи в областта на семантичното търсене, определянето на сходство между изображения и свързани с тях приложения.

6.2.4 Повишаване на мащабируемостта и ефективността

Описаната производствена система демонстрира повишена ефективност и мащабируемост, което подчертава практическите ползи от приложените в нея резултати от научните изследвания, особено при реални широкомащабни внедрявания.

Тестовете за вертикална мащабируемост показаха ефективно използване на наличните хардуерни ресурси. Системата поддържаше високо натоварване на централния процесор (CPU) по време на пикови периоди, без да достига точки на

[15] Набор от данни COCO, обработен с CLIP ViT-L/14: https://huggingface.co/datasets/s-emanuilov/coco-clip-vit-l-14 (DOI: https://www.doi.org/10.57967/hf/3225)

насищане, което е индикация за ефикасно оползотворяване на изчислителната мощ. Използването на оперативната памет се мащабира линейно с обема на информационния масив, като предложеният подход за работа в милиарден мащаб демонстрира ефективно управление на паметта дори при обработка на огромни по обем масиви от векторни данни.

Хоризонталната мащабируемост – ключов фактор за адаптиране към нарастващите обеми от данни и увеличаващото се натоварване от заявки – също показа обещаващи резултати. Решението, базирано на pgvector, демонстрира почти линейно подобрение на производителността при добавяне на допълнителни възли към клъстера. Предложеният подход за хибридно индексиране в милиарден мащаб показа още по-добри характеристики по отношение на мащабируемостта, поддържайки производителност, близка до линейно мащабиране, при увеличаване на броя на възлите. Тази мащабируемост гарантира, че системата може ефективно да се справя с нарастващи обеми данни и увеличаващ се брой потребители чрез лесно добавяне на допълнителни хардуерни ресурси (хоризонтално мащабиране).

Интегрирането на метода LangVec за генериране на интерпретируеми векторни представяния се оказа ценно в практически оперативен контекст. Използването на LangVec улесни по-бързото идентифициране и отстраняване на грешки по време на поддръжката на системата и повиши ефективността при изпълнение на задачи, свързани с оценка на релевантността на резултатите от търсенето.

В заключение, реалното внедряване на резултатите от проведените научни изследвания в тази производствена система потвърждава тяхната практическа приложимост и висока ефективност. Наблюдаваните показатели за производителност, характеристиките на мащабируемост и постигнатото повишаване на ефективността подчертават постигнатия напредък при обработката на широкомащабни задачи, свързани с интензивно търсене и извличане на данни.

6.3 Обобщение

В настоящата глава беше представена оценка и анализ на приложението на разработените в дисертационния труд методи за реализация на софтуерни системи с интензивно използване на данни. Тези методи бяха интегрирани в цялостна система, което послужи за тяхната комплексна оценка (валидиране от второ ниво) в условия, близки до реалните.

Чрез поредица от експериментални изследвания и анализи на конкретни казуси беше демонстрирана ефективността и практическата приложимост на: подхода за хибридно индексиране за търсене по сходство в милиарден мащаб; метода LangVec за подобряване на интерпретируемостта на векторните представяния; и колонно-ориентирания модел на данни за оптимизиране на операциите в системи с интензивно използване на данни.

Интегрирането на тези методи в реална производствена система демонстрира не само тяхната индивидуална практическа приложимост, но и техния синергичен потенциал при изграждането на по-надеждни (стабилни) и ефективни архитектури за системи с интензивно използване на данни.

Заключение

В настоящия дисертационен труд са разгледани многостранните предизвикателства при реализацията на софтуерни системи с интензивно използване на данни, с акцент

върху повишаването на ефективността, мащабируемостта и интерпретируемостта при управлението и анализа на големи информационни масиви. Чрез систематично изследване, обхващащо архитектурни подходи, техники за индексиране, модели за представяне на данни и стратегии за оптимизация, в труда е представен цялостен и съгласуван набор от решения, предназначени да преодолеят ограниченията на съществуващите методи.

Разработването на нов хибриден подход за индексиране адресира проблема с производителността на търсенето по сходство във високоизмерни информационни масиви от порядъка на милиарди записи. Чрез интегриране на плътни векторни вграждания с дискретни атрибути за филтриране в рамките на усъвършенствана структура от тип IVF-Flat, предложеният алгоритъм, оптимизиран за работа с централен процесор (CPU) и използващ дисково пространство, позволява ефективно филтриране и извличане на информация, като същевременно минимизира зависимостта от скъпоструващи ресурси от оперативна памет.

Отчитайки необходимостта от по-добра интерпретируемост на сложни представяния на данни, в настоящия труд е представен LangVec – нов метод за съпоставяне на плътни векторни представяния с лесно четими за човека лексикални репрезентации.

За по-нататъшно оптимизиране на управлението и обработката на данни, в работата са изследвани колонно-ориентираните модели на данни, като е предложен адаптиран дизайн, подходящ за аналитични натоварвания в системи с интензивно използване на данни.

Интегрирането на тези различни техники – хибридния подход за индексиране, метода LangVec и колонно-ориентирания модел на данни – в рамките на реална производствена система за семантично търсене (Similarix) демонстрира техния синергичен потенциал. Това практическо приложение подчертава ползите от резултатите от настоящото изследване в реални условия, демонстрирайки подобрена производителност при изпълнение на заявки, повишена интерпретируемост и по-ефективно управление на големи информационни масиви.

Дисертационният труд обхваща следните теми и изследователски области:

1) **Архитектурни подходи:** Изследване на архитектурните стилове за посрещане на специфичните нужди [8] на системите с интензивно използване на данни.
2) **Алгоритми за индексиране и търсене:** Разработване на ефикасни структури за индексиране и алгоритми за търсене на високоизмерни данни, с акцент върху търсенето на сходство в милиардни масиви от данни.
3) **Техники за интерпретиране на данни:** Създаване на методи за подобряване на интерпретируемостта на сложни представяния на данни, като например преобразуване на плътни вектори в лексикални форми, разбираеми за човека.
4) **Представяне и съхранение на данни:** Изследване на усъвършенствани модели на данни, включително бази данни [42], ориентирани към колони и векторни представяния, за оптимизиране на съхранението, извличането и обработката на големи по обем данни.

Приноси

Научно-приложни приноси

Принос 1. Анализ, методология и оценка на архитектурните подходи за преодоляване на предизвикателствата в системите с интензивно използване на данни (глава 1 и глава 2). Този принос предоставя задълбочен поглед върху приложимостта на различни архитектурни стилове (напр. архитектури със споделени данни, sharding, кеширане) за адресиране на предизвикателствата, свързани с мащабируемостта и производителността при работа с големи обеми от данни. Анализът предлага сравнение на тези стилове спрямо ключовите предизвикателства, характерни за системите с интензивно използване на данни, което улеснява вземането на информирани решения при проектирането на такива системи.

Принос 2. Разработване на нова методика за хибридно индексиране за търсене по сходство в милиарден мащаб (глава 3). Този принос обхваща проектирането на: метод за конструиране на хибридни вектори, който интегрира плътни векторни вграждания с атрибути за филтриране; усъвършенствана индексна структура от тип IVF-Flat, поддържаща многомерно филтриране; и ефективна стратегия за управление на паметта за работа с информационни масиви, чийто обем надхвърля наличната оперативна памет. Предложеният подход дава възможност за ефективно извличане на данни във високоизмерни пространства, като същевременно предлага икономически ефективно решение за мащабно търсене по сходство.

Принос 3. Създаване на методологията LangVec за подобряване на интерпретацията на високоизмерни векторни представяния (глава 4). Това включва разработването на: нова техника за съпоставяне, базирана на персентилно разпределение, за трансформиране на стойностите от векторните компоненти в лексикални представяния (думи/фрази); механизъм за дефиниране на персентилните точки за лексикона с цел балансирано представяне; и метод за обработка на вектори на базата на сегментиране на части (chunks). Методологията подобрява интерпретацията на сложни представяния на данни в системи с интензивно използване на данни, като улеснява по-интуитивното разбиране и анализ на семантичните сходства.

Принос 4. Разработване на колонно-ориентиран модел на данни, оптимизиран за системи с интензивно използване на данни (глава 5). Този принос включва: разработване на подход, базиран на денормализирана схема на данните; алгоритъм за динамично генериране на колонната структура; и техники за оптимизация, осигуряващи висококоростно приемане (ingestion) на данни. Предложеният модел повишава ефективността на съхранение, скоростта на извличане на данни и разширява възможностите за тяхната обработка, особено при аналитични натоварвания и в системи, обработващи милиони записи.

Приложни приноси

Принос 5. Внедряване и разгръщане на производствена софтуерна система, включително:

- Библиотеката LangVec Python, която предоставя практически реализации на методологията за векторна интерпретация (Раздел 6.2);
- Производствена система за семантично търсене (Similarix), интегрираща подхода за хибридно индексиране (Раздел 6.2);

- Webhook система, ориентирана към колони, която демонстрира прилагането на модела на данните (Раздел 5.4);
- Създаване и оценяване на допълнителни инструменти, като например висикоскоростното устройство за сваляне на файлове (gotake), за да се повиши цялостната ефективност на системата (Раздел 6.3).

Принос 6. Разработване на рамки за оценка и практическо валидиране, включително:

- Оценка на производителността и интерпретируемостта на LangVec чрез сравнителни тестове на различни по размер набори от данни и проучване на реален случай с използване на PostgreSQL за откриване на почти дублирани низове (Раздел 4.4);
- Проучване на случай за търсене на сходство в милиарден мащаб, като се използва подмножество на набора от данни LAION-5B, съдържащо 1 милиард вградени изображения, което показва практическата приложимост на подхода за хибридно индексиране за мащабно търсене на сходство с възможност за филтриране (Раздел 3.3);
- Анализ на производителността на модела на данни, ориентиран към колони, в контекста на система за уеб бутони, като се сравняват времената за изпълнение на заявките с традиционните подходи, ориентирани към редове (Раздел 5.4);
- Публикуване на набор от данни с отворен код[16] с вградени данни от подмножество на набора от данни COCO[17] , използван за сравнителен анализ, демонстриран в допълнение B.

Изследванията, представени в тази дисертация, демонстрират последователен подход към многостранните предизвикателства на системите с интензивно използване на данни.

Списък с публикации

1) Simeon Emanuilov, Aleksandar Dimov, Billion-scale Similarity Search Using a Hybrid Indexing Approach with Advanced Filtering, Cybernetics and Information Technologies - Bulgarian Academy of Sciences, 2024, ISSN (print):1311-9702, ISSN (online):1314-4081 [43]
2) Simeon Emanuilov, Aleksandar Dimov, Lexical Representation of Dense Numerical Vectors: Introducing LangVec, Mathematics and Informatics - Az-buki, 2024, ISSN (print):1310–2230, ISSN (online):1314–8532, doi:10.53656/math2024-3-1-lex [6]
3) Simeon Emanuilov, Aleksandar Dimov, Column-oriented data model for data-intensive systems, Proceedings in the International Scientific Conference “Computer Science' 2022”, Publisher: IEEE Xplore Digital Library, 2022, ISSN (online):978-1-6654-9777-0, doi:10.1109/COMSCI55378.2022.9912610 [44]

[16] Набор от данни COCO, обработен с CLIP ViT-L/14: https://huggingface.co/datasets/s-emanuilov/coco-clip-vit-l-14
[17] Набор от данни COCO (Common Objects in Context): https://cocodataset.org/#home

Списък с презентации

1) **Design of a billion-scale semantic search system**, FMI Spring Scientific Session, 2024, https://www.fmi.uni-sofia.bg/bg/proletna-nauchna-sesiya-na-fmi-2024

2) **Challenges in building a billion-scale semantic search system on a CPU server**, SUMMIT, Annual conference, 2024, https://www.uni-sofia.bg/index.php/novini/kalendar/p_rva_godishna_konferenciya_na_proekt_summit_na_sofijskiya_universitet

3) **Software design patterns for building scalable Machine Learning systems**, FMI Spring Scientific Session, 2023, https://www.fmi.uni-sofia.bg/bg/proletna-nauchna-sesiya-na-fmi-2023

4) **Machine learning vs traditional software systems**, AI-BEST project workshop, 09.12.2023, https://www.uni-sofia.bg/index.php/novini/novini_i_s_bitiya/vtori_seminar_po_proekta_ai_best_ai_and_big_data_for_education_software_and_information_technologies

5) **Architectural approaches to overcome challenges in the development of data-intensive systems**, 13th International Conference on Applied Human Factors and Ergonomics, 2022, https://ahfe.org/files/AHFE2022_FinalProgram.pdf

6) **Task scheduling in distributed data-intensive systems**, FMI Spring Scientific Session, 2022, https://www.fmi.uni-sofia.bg/bg/proletna-nauchna-sesiya-na-fmi-2022

7) **Column-oriented data model for data-intensive systems**, 2022 10th International Scientific Conference COMPUTER SCIENCE, 2022, https://ieeexplore.ieee.org/xpl/conhome/9912441/proceeding

8) **Methods for implementation of data intensive software systems**, FMI Spring Scientific Session, 2021, https://www.fmi.uni-sofia.bg/bg/proletna-nauchna-sesiya-na-fmi-2021

Библиография

[1] H. Allioui и Y. Mourdi, „Exploring the full potentials of IoT for better financial growth and stability: A comprehensive survey.,“ *Sensors,* том 23, № 19, p. 8015, 2023.

[2] E. Siow, T. Tiropanis и W. Hall, „Analytics for the internet of things: A survey,“ *ACM computing surveys (CSUR),* том 51, № 4, pp. 1-36, 2018.

[3] L. Wang и A. A. Cheryl , „Machine learning in big data,“ *International Journal of Mathematical, Engineering and Management Sciences,* pp. 52-61, 2016.

[4] Y. Demchenko, C. Laat и P. Membrey, „Defining architecture components of the Big Data Ecosystem,“ в *International Conference on Collaboration Technologies and Systems*, Minneapolis, 2014.

[5] T. Kosar, S. Vazhkudai, A. Butt и X. Ma, „Distributed Storage Systems for Data Intensive Computing,“ в *Data Intensive Distributed Computing*, 2012, p. 23.

[6] S. Emanuilov и A. Dimov, „Lexical representation of dense numerical vectors: Introducing LangVec,“ *Mathematics & Informatics,* том 67, № 3, 2024.

[7] C. P. Chen и Z. Chun-Yang, „Data-intensive applications, challenges, techniques and technologies: A survey on Big Data.,“ *Information sciences,* том 275, pp. 314-347, 2014.

[8] M. Kleppmann, Designing data-intensive applications: The big ideas behind reliable, scalable, and maintainable systems., O'Reilly Media, Inc, 2017.

[9] P. Pääkkönen и D. Pakkala, „Reference architecture and classification of technologies, products and services for big data systems,“ *Big data research,* том 2, № 4, pp. 166-186, 2015.

[10] U. M. Graetsch, H. Khalajzadeh, M. Shahin, R. Hoda и J. Grundy, „Dealing with data challenges when delivering data-intensive software solutions,“ *IEEE Transactions on Software Engineering,* pp. 1-23, 2023.

[11] C. Huyen, Designing machine learning systems, Sebastopol: O'Reilly Media, Inc., 2022.

[12] P. Nagar, S. Boruah, A. K. Bhoi, A. Patel, J. Sarda и P. Darjij, „Emerging VLSI Technologies for High performance AI and ML Applications,“ в *International Conference on Advancements in Smart, Secure and Intelligent Computing (ASSIC)*, Bhubaneswar, 2024.

[13] M. Capra, B. Bussolino, A. Marchisio, G. Masera, M. Martina и M. Shafique, „Hardware and software optimizations for accelerating deep neural networks: Survey of current trends, challenges, and the road ahead,“ *IEEE Access,* том 8, pp. 225134-225180, 2020.

[14] L. Baier, F. Jöhren и S. Seebacher, „Challenges in the Deployment and Operation of Machine Learning in Practice,“ в *European Conference on Information Systems (ECIS)*, Stockholm, 2019.

[15] A. Serban, K. van der Blom, H. Hoos и J. Visser, „Software engineering practices for machine learning—Adoption, effects, and team assessment,“ *Journal of Systems and Software,* том 209, p. 111907, 2024.

[16] D. Sculley, G. Holt, D. Golovin, E. Davydov, T. Phillips, D. Ebner, V. Chaudhary, M. Young, J.-F. Crespo и D. Dennison, „Hidden technical debt in machine learning systems,“ *Advances in neural information processing systems,* том 28, 2015.

[17] E. Breck, S. Cai, E. Nielsen, M. Salib и D. Sculley, „The ML test score: A rubric for ML production readiness and technical debt reduction,“ *EEE international conference on big data (big data),* pp. 1123-1132, 2017.

[18] S. Amershi, A. Begel, C. Bird, R. DeLine, H. Gall, E. Kamar, N. Nagappan, B. Nushi и T. Zimmermann, „Software engineering for machine learning: A case study,“ в *EEE/ACM 41st International Conference on Software Engineering: Software Engineering in Practice (ICSE-SEIP)*, Montreal, 2019.

[19] I. F. Smith, S. Saitta, S. Ravindran и P. Kripakaran, „Challenges of data interpretation,“ в *8th SAMCO Workshop*, 2006.

[20] K. Echihabi, K. Zoumpatianos и T. Palpanas, „High-dimensional similarity search for scalable data science,“ в *IEEE 37th International Conference on Data Engineering (ICDE)*, Chania, 2021.

[21] J. Johnson, M. Douze и H. Jégou, „Billion-scale similarity search with GPUs,“ *IEEE Transactions on Big Data,* том 7, № 3, pp. 535-547, 2019.

[22] I. T. Jolliffe, Principal component analysis for special types of data, New York: Springer, 2002.

[23] L. Van der Maaten и G. Hinton, „Visualizing data using t-SNE,“ *Journal of machine learning research,* том 9, № 11, 2008.

[24] L. McInnes, J. Healy и J. Melville, *Umap: Uniform manifold approximation and projection for dimension reduction,* arXiv preprint arXiv:1802.03426, 2018.

[25] J. Wang, H. Tao Shen, J. Song и J. Ji, *Hashing for similarity search: A survey,* arXiv preprint arXiv:1408.2927., 2014.

[26] T. Tran, D. M. Herzig и G. Ladwig, „SemSearchPro–Using semantics throughout the search process," *Journal of Web Semantics,* том 9, № 4, pp. 349-364, 2011.

[27] M. Gärtner, A. Rauber и H. Berger, „Bridging structured and unstructured data via hybrid semantic search and interactive ontology-enhanced query formulation," *Knowledge and information systems,* том 41, pp. 761-792, 2014.

[28] A. F. Da Costa, M. A. Domingues, S. O. Rezende и M. G. Manzato, „Improving personalized ranking in recommender systems with multimodal interactions," в *2014 IEEE/WIC/ACM International Joint Conferences on Web Intelligence (WI) and Intelligent Agent Technologies (IAT), vol. 1, pp. 198-204. IEEE*, 2014.

[29] M. Köppen, „The curse of dimensionality," *5th online world conference on soft computing in industrial applications (WSC5),* 2000.

[30] J. Subramanya, F. D. Suhas, H. V. Simhadri, R. Krishnawamy и R. Kadekodi, „DiskANN: Fast accurate billion-point nearest neighbor search on a single node," в *Advances in Neural Information Processing Systems 32*, Vancouver, 2019.

[31] S. Gollapudi, N. Karia, V. Sivashankar, R. Krishnaswamy, N. Begwani, S. Raz и Y. Lin, „Filtered-DiskANN: Graph algorithms for approximate nearest neighbor search with filters," в *Proceedings of the ACM Web Conference*, 2023.

[32] H. Jegou, M. Douze и C. Schmid, „Product quantization for nearest neighbor search," *IEEE transactions on pattern analysis and machine intelligence,* том 33, № 1, pp. 117-128, 2010.

[33] Y. A. Malkov и D. A. Yashunin, „Efficient and robust approximate nearest neighbor search using hierarchical navigable small world graphs," *IEEE transactions on pattern analysis and machine intelligence,* том 42, № 4, pp. 824-836, 2018.

[34] Z.-b. Wu и J.-q. Yu, „Vector quantization: a review," *Frontiers of Information Technology & Electronic Engineering,* том 20, № 4, pp. 507-524, 2019.

[35] D. Baranchuk, A. Babenko и Y. Malkov, „Revisiting the inverted indices for billion-scale approximate nearest neighbors," в *Proceedings of the European Conference on Computer Vision (ECCV)*, Munich, 2018.

[36] D. Gunning и D. Aha, „DARPA's explainable artificial intelligence (XAI) program," *AI magazine,* том 40, № 2, pp. 44-58, 2019.

[37] M. T. Ribeiro, S. Singh и C. Guestrin, „Why should i trust you?" Explaining the predictions of any classifier," в *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, San Francisco, 2016.

[38] C. Schuhmann, R. Beaumont, R. Vencu, C. Gordon, R. Wightman, M. Cherti и T. Coombes, „Laion-5b: An open large-scale dataset for training next generation image-text models," в *Advances in Neural Information Processing Systems 35*, 2022.

[39] D. Zuras, M. Cowlishaw, A. Aiken, M. Applegate, D. Bailey, S. Bass и D. Bhandarkar, „IEEE standard for floating-point arithmetic," IEEE Std 754, 2008.

[40] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal и G. Sastry, „Learning transferable visual models from natural language supervision," в *International conference on machine learning, pp. 8748-8763. PMLR*, 2021.

[41] R. Girdhar, A. El-Nouby, Z. Liu, M. Singh, K. V. Alwala, A. Joulin и I. Misra, „ImageBind: One embedding space to bind them all," в *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 15180-15190*, Vancouver, 2023.

[42] N. Stratos, F. Idreos, N. Groffen, M. Kersten, S. Mullender и S. Manegol, „MonetDB: Two decades of research in column-oriented database architectures.," *IEEE Data Engineering,* том 40, 2012.

[43] S. Emanuilov и A. Dimov, „Billion-scale Similarity Search Using a Hybrid Indexing Approach with Advanced Filtering," 2024.

[44] S. Emanuilov и A. Dimov, „Column-oriented data model for data-intensive systems," в *Proceedings in the International Scientific Conference "Computer Science' 2022", Publisher:IEEE Xplore Digital Library*, 2022.

[45] C. Dubnicki, E. Kruus и C. Ungureanu, „Methods and systems for quick and efficient data management and/or processing“. U.S. Патент 8,214,517, 3 July 2012.

[46] P. R. Chelliah, S. Harihara и M. Anupama, Architectural Patterns: Uncover essential patterns in the most indispensable realm of enterprise architecture., Birmingham: Packt Publishing Ltd, 2017.

[47] S. M. Levin, „Unleashing real-time analytics: A comparative study of in-memory computing vs. traditional disk-based systems.,“ *Brazilian Journal of Science,* том 3, № 5, pp. 30-39, 2024.

[48] J. J. Pan, W. Jianguo и L. Guoliang, „Vector Database Management Techniques and Systems,“ в *Companion of the 2024 International Conference on Management of Data.*, New York, 2024.

[49] R. Bryant, „Data-Intensive Supercomputing: The case for DISC,“ Carnegie Mellon University, Pittsburgh, 2007.

[50] F. Al-Hawari, „Software design patterns for data management features in web-based information systems,“ *Journal of King Saud University-Computer and Information Sciences,* том 34, № 10, pp. 10028-10043, 2022.

[51] H. Muccini и K. Vaidhyanathan, „Software architecture for ML-based systems: What exists and what lies ahead,“ в *AI Engineering - Software Engineering for AI (WAIN) Workshop at ICSE 2021*, Madrid, 2021.

[52] L. E. Lwakatare, A. Raj, J. Bosch, H. H. Olsson и I. Crnkovic, „A taxonomy of software engineering challenges for machine learning systems: An empirical investigation,“ в *Agile Processes in Software Engineering and Extreme Programming: 20th International Conference*, Montreal, 2019.

[53] J. Bosch, H. Olsson и I. Crnkovic, „Engineering AI Systems: A Research Agenda,“ в *Artificial Intelligence Paradigms for Smart Cyber-Physical Systems*, 2021.

[54] N. Humbatova, G. Jahangirova, G. Bavota, V. Riccio, A. Stocco и P. Tonella, „Taxonomy of real faults in deep learning systems,“ в *Proceedings of the ACM/IEEE 42nd international conference on software engineering*, Seoul, 2020.

[55] V. Lenarduzzi, F. Lomio, S. Moreschini, D. Taibi и D. Tamburri, „Software quality for ai: Where we are now? in Software Quality: Future Perspectives on Software Engineering Quality,“ в *13th International Conference, SWQD*, Vienna, 2021.

[56] B. Sturmfels, „Voronoi Cells,“ UC Berkeley, 7 February 2023. [Онлайн]. Available: https://math.berkeley.edu/~bernd/wednesday.pdf. [Отваряно на 5 10 2024].

[57] K. Yang, H. Wang, M. Du, Z. Wang, Z. Tan и Y. Xiao, „Hierarchical Link and Code: Efficient Similarity Search for Billion-Scale Image Sets,“ *PG (Short Papers, Posters, and Work-in-Progress Papers),* pp. 81-86, 2021.

[58] A. Singh, S. Jayaram Subramanya, R. Krishnaswamy и H. S. Vardhan, *FreshDiskANN: A Fast and Accurate Graph-Based ANN Index for Streaming Similarity Search,* arXiv preprint arXiv:2105.09613, 2021.

[59] N. Sundaram, A. Turmukhametova, N. Satish, T. Mostak, P. Indyk, S. Madden и P. Dubey, „Streaming similarity search over one billion tweets using parallel locality-sensitive hashing,“ в *Proceedings of the VLDB Endowment 6, no. 14* , 2013.

[60] L. Faubel и K. Schmid, *MLOps: A Multiple Case Study in Industry 4.0,* arXiv preprint arXiv:2407.09107, 2024.

[61] H. Washizaki, H. Uchida, F. Khomh и Y.-G. Guéhéneuc, „Studying software engineering patterns for designing machine learning systems,“ в *10th International Workshop on Empirical Software Engineering in Practice (IWESEP)*, 2019.

[62] V. Lakshmanan, S. Robinson и M. Munn, Machine learning design patterns, Sebastopol: O'Reilly Media, 2020.

[63] C. O. S. Sorzano, J. Vargas и A. P. Montano, *A survey of dimensionality reduction techniques,* arXiv preprint arXiv:1403.2877, 2014.

[64] G. T. Reddy, M. P. K. Reddy, K. Lakshmanna, R. Kaluri, D. Singh Rajput, G. Srivastava и T. Baker, „Analysis of dimensionality reduction techniques on big data,“ *IEEE Access,* том 8, pp. 54776-54788, 2020.

[65] X. He и P. Niyogi, „Locality preserving projections,“ *Advances in neural information processing systems,* том 16, 2003.

[66] A. Wang, S. Zhao, J. Liu, J. Yang, L. Liu и G. Chen, „Locality adaptive preserving projections for linear dimensionality reduction,“ *Expert Systems with Applications,* том 151, p. 113352, 2020.

[67] S. Lundberg, *A unified approach to interpreting model predictions,* arXiv preprint arXiv:1705.07874, 2017.

[68] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh и D. Batra, „Grad-cam: Visual explanations from deep networks via gradient-based localization," в *Proceedings of the IEEE international conference on computer vision*, 2017.

[69] B. Kim, M. Wattenberg, J. Gilmer, C. Cai, J. Wexler и F. Viegas, „Interpretability beyond feature attribution: Quantitative testing with concept activation vectors (TCAV)," в *International conference on machine learning*, 2018.

[70] S. Wachter, B. Mittelstadt и C. Russell, „Counterfactual explanations without opening the black box: Automated decisions and the GDPR," *Harvard journal of law & technology,* том 31, № 2, pp. 841-887, 2018.

[71] S. Verma, V. Boonsanong, M. Hoang, K. Hines, J. Dickerson и C. Shah, „Counterfactual explanations and algorithmic recourses for machine learning: A review.," *ACM Computing Surveys,* том 56, 2020.

[72] A. Dimov, S. Emanuilov, B. Bontchev, Y. Dankov и T. Papapostolu, „Architectural approaches to overcome challenges in the development of data-intensive systems," в *Human Factors in Software and Systems Engineering*, New York, 2022.

[73] M. S. Khorshidi, N. Yazdanjue, H. Gharoun, D. Yazdani, M. R. Nikoo, F. Chen и A. H. Gandomi, *Semantic-Preserving Feature Partitioning for Multi-View Ensemble Learning,* arXiv preprint arXiv:2401.06251, 2024.

[74] B. Gedik, „Auto-tuning similarity search algorithms on multi-core architectures," *International Journal of Parallel Programming,* том 41, № 5, pp. 595-620, 2013.

[75] P. Zhang, Z. Liu, S. Xiao, Z. Dou и J. Yao, „Hybrid Inverted Index Is a Robust Accelerator for Dense Retrieval," в *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, Singapore, 2023.

[76] G. Gupta, T. Medini, A. Shrivastava и A. J. Smola, „BLISS: A billion scale index using iterative re-partitioning.," в *28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, Washington DC, 2022.

[77] H. Li, J. Wang, Y. Zheng, L. Wang, W. Zhang и H.-W. Shen, „Compressing and interpreting word embeddings with latent space regularization and interactive semantics probing," *Information Visualization,* том 22, № 1, pp. 52-68, 2023.

[78] J. Walmsley, „Artificial intelligence and the value of transparency," *AI & society,* том 36, № 2, pp. 585-595, 2021.

[79] Y. Ke, R. Sukthankar, L. Huston, Y. Ke и R. Sukthankar, „Efficient near-duplicate detection and sub-image retrieval," в *ACM multimedia, vol. 4, no. 1*, 2004.

[80] P. Indyk и R. Motwani, „Approximate nearest neighbors: towards removing the curse of dimensionality," в *Proceedings of the thirtieth annual ACM symposium on Theory of computing*, 1998.

[81] A. Gionis, P. Indyk и R. Motwani, „Similarity search in high dimensions via hashing," *VLDB,* том 99, № 6, pp. 518-529, 1999.

[82] T. Mikolov, *Efficient estimation of word representations in vector space,* arXiv preprint arXiv:1301.3781, 2013.

[83] J. Pennington, R. Socher и C. D. Manning, „Glove: Global vectors for word representation," в *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, Doha, 2014.

[84] J. Devlin, *BERT: Pre-training of deep bidirectional transformers for language understanding,* arXiv preprint arXiv:1810.04805, 2018.

[85] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei и I. Sutskever, *Language models are unsupervised multitask learners,* OpenAI blog 1, no. 8, 2019.

[86] M. Mars, „From word embeddings to pre-trained language models: A state-of-the-art walkthrough," *Applied Sciences,* том 12, № 17, p. 8805, 2022.

[87] K. Clark, *What Does Bert Look At? An Analysis of Bert's Attention,* arXiv preprint arXiv:1906.04341, 2019.

[88] A. Conneau, G. Kruszewski, G. Lample, L. Barrault и M. Baroni, *What you can cram into a single vector: Probing sentence embeddings for linguistic properties,* arXiv preprint arXiv:1805.01070, 2018.

[89] E. Voita, D. Talbot, F. Moiseev, R. Sennrich и I. Titov, *Analyzing multi-head self-attention: Specialized heads do the heavy lifting, the rest can be pruned,* arXiv preprint arXiv:1905.09418, 2019.

[90] A. Ghorbani, J. Wexler, J. Y. Zou и B. Kim, „Towards automatic concept-based explanations," в *Advances in neural information processing systems 32*, 2019.

[91] X. Wei, M. J. Gales и K. M. Knill, „Analysing bias in spoken language assessment using concept activation vectors," в *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Toronto, 2021.

[92] K. Chard, S. Tuecke и I. Foster, „Efficient and secure transfer, synchronization, and sharing of big data," *IEEE Cloud Computing,* том 1, № 3, pp. 46-55, 2014.

[93] E. Manogar и S. Abirami, „A study on data deduplication techniques for optimized storage," в *Sixth International Conference on Advanced Computing (ICoAC), pp. 161-166. IEEE*, 2014.

[94] V. Sarkar, W. Harrod и A. E. Snavely, „Software challenges in extreme scale systems," *Journal of Physics: Conference Series,* том 180, № 1, p. 012045, 2009.

[95] R.-W. Bello и F. N. Otobo, „Hardware/software interoperability and single point vulnerability problems of internet of things multiple systems: Causes, solution and societal adoption," *Asian Journal of Mathematical Sciences,* том 2, № 2, 2018.

[96] D. Preuveneers и W. Joosen, „Automated configuration of NoSQL performance and scalability tactics for data-intensive applications," *Informatics,* том 7, № 3, p. 29, 2020.

[97] B. M. Abdelhafiz и M. Elhadef, „Sharding database for fault tolerance and scalability of data," в *2021 2nd International Conference on Computation, Automation and Knowledge Management (ICCAKM) IEEE*, 2021.

[98] A. Krechowicz, S. Deniziak и G. Łukawski, „Highly scalable distributed architecture for NoSQL datastore supporting strong consistency," *IEEE Access,* том 9, pp. 69027-69043, 2021.

[99] Q. Wei, B. Li, W. Chang, Z. Jia, Z. Shen и Z. Shao, „A survey of blockchain data management systems," *ACM Transactions on Embedded Computing Systems (TECS),* том 21, № 3, pp. 1-28, 2022.

[100] A. Ayyagiri, P. Kirupa, G. Pandian и P. Goel, „Efficient Data Migration Strategies in Sharded Databases," *Journal of Quantum Science and Technology,* том 1, № 2, pp. 72-87, 2024.

[101] B. Sena, L. Garcés, A. Paula Allian и E. Yumi Nakagawa, „Investigating the applicability of architectural patterns in big data systems," в *Proceedings of the 25th conference on pattern languages of programs, pp. 1-15*, 2018.

[102] C. Wulf, *Efficient Engineering and Execution of Pipe-and-Filter Architectures,* PhD diss, Faculty of Engineering, Kiel University, 2019.

[103] D. Bonino и L. De Russis, „Complex event processing for city officers: A filter and pipe visual approach," *IEEE Internet of Things Journal,* том 5, № 2, pp. 775-783, 2017.

[104] C. Wulf, C. Claus Wiechmann и W. Hasselbring, „Increasing the throughput of pipe-and-filter architectures by integrating the task farm parallelization pattern," в *19th International ACM SIGSOFT Symposium on Component-Based Software Engineering (CBSE)*, Venice, 2016.

[105] C. Wulf, W. Hasselbring и J. Ohlemacher, „Parallel and generic pipe-and-filter architectures with TeeTime," в *IEEE International Conference on Software Architecture Workshops (ICSAW), pp. 290-293*, Gothenburg, 2017.

[106] I. Gorton, A. Wynne, Y. Liu и J. Yin, „Components in the Pipeline," *IEEE software,* том 28, № 3, pp. 34-40, 2011.

[107] M. Li, F. Ye, M. Kim, H. Chen и H. Lei, „A scalable and elastic publish/subscribe service," в *IEEE International Parallel & Distributed Processing Symposium, pp. 1254-1265*, Alaska, 2011.

[108] S. Oh, J.-H. Kim и G. Fox, „Real-time performance analysis for publish/subscribe systems," *Future Generation Computer Systems,* том 26, № 3, pp. 318-323, 2010.

[109] H. Zhang, A. Sharma, Y. Zhang, S. Ganguly, S. Bhatnagar и R. Izmailov, „A Scalable Publish/Subscribe Broker Network Using Active Load Balancing," в *Future Internet Services and Service Architectures*, Gistrup, River Publishers, 2022, pp. 317-337.

[110] T. Chang, S. Duan, H. Meling, S. Peisert и H. Zhang, „P2S: a fault-tolerant publish/subscribe infrastructure," в *Proceedings of the 8th ACM International Conference on Distributed Event-Based Systems, pp. 189-197.*, 2014.

[111] R. Baldoni, S. Bonomi, M. Platania и L. Querzoni, „Dynamic message ordering for topic-based publish/subscribe systems,“ в *IEEE 26th International Parallel and Distributed Processing Symposium*, 2012.

[112] S. Bhola, R. Strom, S. Bagchi, Y. Zhao и J. Auerbach, „Exactly-once delivery in a content-based publish-subscribe system,“ в *Proceedings International Conference on Dependable Systems and Networks, pp. 7-16. IEEE*, 2002.

[113] Y. Zhao и J. Wu, „Building a reliable and high-performance content-based publish/subscribe system,“ *Journal of Parallel and Distributed Computing,* том 73, № 4, pp. 371-382, 2013.

[114] P. T. Eugster, P. A. Felber, R. Guerraoui и A.-M. Kermarrec, „The many faces of publish/subscribe,“ *ACM computing surveys (CSUR),* том 35, № 2, pp. 114-131, 2003.

[115] L. Saino, *On the design of efficient caching systems,* London: PhD diss., UCL (University College London), 2015.

[116] R. Koller, L. Marmol, R. Rangaswami, S. Sundararaman, N. Talagala и M. Zhao, „Write policies for host-side flash caches,“ в *11th USENIX Conference on File and Storage Technologies (FAST 13), pp. 45-58*, 2013.

[117] T. Mann, A. Birrell, A. Hisgen, C. Jerian и G. Swart, „A coherent distributed file cache with directory write-behind,“ *ACM Transactions on Computer Systems (TOCS),* том 12, № 2, pp. 123-164, 1994.

[118] J. Zhang и E. Yeh, *LOAM: Low-latency Communication, Caching, and Computation Placement in Data-Intensive Computing Networks,* arXiv preprint arXiv:2403.15927, 2024.

[119] Q. Luo, S. Krishnamurthy, C. Mohan, H. Pirahesh, H. Woo, B. G. Lindsay и J. F. Naughton, „Middle-tier database caching for e-business,“ в *Proceedings of the 2002 ACM SIGMOD international conference on Management of data, pp. 600-611*, 2002.

[120] M. Pietro, D. Carra и S. Migliorini, „In-memory caching for multi-query optimization of data-intensive scalable computing workloads,“ в *Proceedings of the Workshops of the EDBT/ICDT 2019 Joint Conference (EDBT/ICDT 2019), pp. 1-8.*, 2019.

[121] I. Raicu, . I. T. Foster, Y. Zhao, P. Little, C. M. Moretti, A. Chaudhary и D. Thain, „The quest for scalable support of data-intensive workloads in distributed systems,“ в *Proceedings of the 18th ACM international symposium on High performance distributed computing, pp. 207-216.*, 2009.

[122] H. Shukur, S. Zeebaree, R. Zebari, O. Ahmed, L. Haji и D. Abdulqader, „Cache coherence protocols in distributed systems,“ *Journal of Applied Science and Technology Trends,* том 1, № 2, pp. 92-97, 2020.

[123] A. Manjhi, A. Ailamaki, B. M. Maggs, T. C. Mowry, C. Olston и A. Tomasic, „Simultaneous scalability and security for data-intensive web applications,“ в *Proceedings of the 2006 ACM SIGMOD international conference on Management of data, pp. 241-252.*, 2006.

[124] C. Chen, L. Nagel, L. Cui и F. P. Tso, „S-cache: Function caching for serverless edge computing,“ в *Proceedings of the 6th International Workshop on Edge Systems, Analytics and Networking, pp. 1-6.*, Rome, 2023.

[125] H. Zhang, G. Chen, B. C. Ooi, K.-L. Tan и M. Zhang, „In-memory big data management and processing: A survey,“ *IEEE Transactions on Knowledge and Data Engineering,* том 27, № 7, pp. 1920-1948, 2015.

[126] L. Yujian и L. Bo, „A normalized Levenshtein distance metric,“ *IEEE transactions on pattern analysis and machine intelligence,* том 29, № 6, pp. 1091-1095, 2007.

[127] T. Georgiou, Y. Liu, W. Chen и M. Lew, „A survey of traditional and deep learning-based feature descriptors for high dimensional data in computer vision,“ *International Journal of Multimedia Information Retrieval,* том 9, pp. 135-170, 2020.

[128] W. J. Murdoch, C. Singh, K. Kumbier, R. Abbasi-Asl и B. Yu, „Definitions, methods, and applications in interpretable machine learning,“ *Proceedings of the National Academy of Sciences,* том 116, № 44, pp. 22071-22080, 2019.

[129] A. Strehl и J. Ghosh, „Relationship-based clustering and visualization for high-dimensional data mining,“ *INFORMS Journal on Computing,* том 15, № 2, pp. 208-230, 2003.

[130] L. Gao, J. Song, X. Liu, J. Shao, J. Liu и J. Shao, „Learning in high-dimensional multimedia data: the state of the art,“ *Multimedia Systems,* том 23, pp. 303-313, 2017.

[131] W. Jia, M. Sun, J. Lian и S. Hou, „Feature dimensionality reduction: A review,“ *Complex & Intelligent Systems,* том 8, № 3, pp. 2663-2693, 2022.

[132] S. Ayesha, M. K. H. Hanif и R. Talib, „Overview and comparative study of dimensionality reduction techniques for high dimensional data,“ *Information Fusion,* том 59, pp. 44-58, 2020.

[133] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár и C. L. Zitnick, „Microsoft COCO: Common objects in context," в *Computer Vision–ECCV 2014: 13th European Conference, Proceedings, Part V 13, pp. 740-755*, Zurich, 2014.

[134] S. Bansal, „Assessment of various simulation models incorporating queuing concept," *Journal of Soft Computing Paradigm,* том 4, № 3, pp. 121-128, 2022.

[135] A. Huynh, H. A. Chaudhari, E. Terzi и M. Athanassoulis, „Towards flexibility and robustness of LSM trees," *The VLDB Journal,* pp. 1-24, 2024.

[136] T. Pelkonen, S. Franklin, J. Teller, P. Cavallaro, Q. Huang, J. Meza и K. Veeraraghavan, „Gorilla: A fast, scalable, in-memory time series database," *Proceedings of the VLDB Endowment,* том 8, № 12, pp. 1816-1827, 2015.

[137] S. Tang, B.-S. Lee и B. He, „Fair resource allocation for data-intensive computing in the cloud," *IEEE Transactions on Services Computing,* том 11, № 1, pp. 20-33, 2016.

[138] I. Zacharov, R. Arslanov, M. Gunin, D. Stefonishin, A. Bykov, S. Pavlov, O. Panarin, A. Maliutin, S. Rykovanov и M. Fedorov, „"Zhores"—Petaflops supercomputer for data-driven modeling, machine learning and artificial intelligence installed in Skolkovo Institute of Science and Technology," *Open Engineering,* том 9, № 1, pp. 512-520, 2019.

[139] H. Sheshadri, S. Vijayakumar, A. Jacob и A. Jaiswal, *Augmented Memory Computing: Dynamically Augmented SRAM Storage for Data Intensive Applications,* arXiv preprint arXiv:2109.03022, 2021.

[140] Y. Wang и Z. Zhao, „Decentralized workflow management on software defined infrastructures," в *2020 IEEE World Congress on Services (SERVICES), pp. 265-268*, 2020.

[141] K. Kara, C. Hagleitner, D. Diamantopoulos, D. Syrivelis и G. Alonso, „High bandwidth memory on FPGAs: A data analytics perspective," в *2020 30th International Conference on Field-Programmable Logic and Applications (FPL), pp. 1-8. IEEE*, 2020.

[142] Z. Li, „On a factorial knowledge architecture for data science-powered software engineering," в *Proceedings of the 2020 4th International Conference on Software and e-Business, pp. 20-24*, 2020.

[143] S. Nayak, R. Patgiri и T. D. Singh, „Big Computing: Where are we heading?," *ICST Transactions on Scalable Information Systems ,* pp. 1-10, 2020.

[144] I. Suriarachchi, S. Withana и B. Plale, „Big provenance stream processing for data intensive computations," в *2018 IEEE 14th International Conference on e-Science (e-Science), pp. 245-255. IEEE*, 2018.

[145] S. Palkar, J. Thomas, D. Narayanan, P. Thaker, R. Palamuttam, P. Negi и A. Shanbhag, „Evaluating end-to-end optimization for data analytics applications in weld," *Proceedings of the VLDB Endowment,* том 11, № 9, pp. 1002-1015, 2018.

[146] M. Alhayan и H. Abuhassna, „A Comprehensive Bibliometric Analysis of Cybersecurity Trends on Cloud Computing," *International Journal of Academic Research in Business and Social Sciences,* том 14, № 1, 2024.

[147] D. M. Uma и G. M. Gandhi, „Wordnet and Ontology Based Query Expansion for Semantic Information Retrieval in Sports Domain," *Journal of Computer Science,* том 11, № 2, pp. 361-371, 2015.

[148] E. Inan, P. Thompson, T. Yates и S. Ananiadou, „HSEarch: Semantic search system for workplace accident reports," в *Advances in Information Retrieval: 43rd European Conference on IR Research, ECIR 2021, Virtual Event, March 28–April 1, 2021, Proceedings, Part II 43, pp. 514-519. Springer International Publishing*, 2021.

[149] D. W. Embley, S. W. Liddle, D. W. Lonsdale, J. S. Park, B.-J. Shin и A. J. Zitzelberger, „Cross-language hybrid keyword and semantic search," в *Conceptual Modeling: 31st International Conference ER 2012, Florence, Italy, October 15-18, 2012. Proceedings 31, pp. 190-203. Springer Berlin Heidelberg,*, Berlin, 2012.

[150] B. Wilder, Cloud architecture patterns: Using Microsoft Azure, Sebastopol: O'Reilly Media, Inc., 2012.

[151] M. Kaloudis, „Evolving Software Architectures from Monolithic Systems to Resilient Microservices: Best Practices, Challenges and Future Trends," *International Journal of Advanced Computer Science & Applications,* том 15, № 9, 2024.

ДЕКЛАРАЦИЯ

от Симеон Стоичков Емануилов,
задочен докторант към катедра “Софтуерни Технологии”

Във връзка с: Дисертационен труд за придобиване на образователна и научна степен “Доктор” на тема “Методи за реализация на софтуерни системи за обработка на големи данни”.

Декларирам, че:

1. Представеният дисертационен труд е мое собствено дело и в него не са използвани пряко или косвено чужди текстове, без същите да бъдат надлежно цитирани.

2. Никоя част от дисертационния труд не е в нарушение на авторските права на институция или личност.

3. Научните трудове и цитиранията като доказателства по наукометричните показатели в представената по настоящата процедура “Справка за изпълнение на минималните национални изисквания” по чл. 26, ал. 2 и 3 на ЗРАС на РБ за придобиване на ОНС „Доктор" не са използвани от мен за придобиване на друга научна степен или за заемане на академична длъжност по същото или друго професионално направление.

Известна ми е наказателната отговорност, която нося по чл. 313 от Наказателния кодекс за деклариране на неверни данни.

Дата: 24.05.2025

Подпис:

Симеон Емануилов