

УДК 004.8RAG:005
ББК 16.6+65.291с
Д86

Душкин Р. В.

Д86 RAG-системы: от теории к практике. – М.: ДМК Пресс, 2025. – 286 с.: ил.

ISBN 978-5-93700-429-1

Книга рассказывает о том, как научить большие языковые модели работать с актуальными корпоративными данными при помощи систем класса RAG (Retrieval-Augmented Generation), способных получать доступ к внешним базам знаний и генерировать точные, актуальные ответы на поисковые запросы сотрудников и клиентов компаний.

Издание предназначено разработчикам интеллектуальных систем, ML-инженерам, архитекторам корпоративных решений и всем, кто профессионально работает с генеративным искусственным интеллектом, а также тем, кто стремится внедрить передовые технологии в свой бизнес.

УДК 004.8RAG:005
ББК 16.6+65.291с

Все права защищены. Любая часть этой книги не может быть воспроизведена в какой бы то ни было форме и какими бы то ни было средствами без письменного разрешения владельцев авторских прав.

ISBN 978-5-93700-429-1

© Душкин Р. В., 2025
© Оформление, издание,
ДМК Пресс, 2025

В сфере искусственного интеллекта нет ничего более парадоксального, чем скорость устаревания знаний. За то время, пока готовится к печати очередная техническая книга, появляются новые фреймворки, алгоритмы и подходы, которые кардинально меняют ландшафт технологий. RAG-системы не исключение – эта область развивается настолько стремительно, что любая попытка зафиксировать «окончательное» знание обречена на устаревание ещё до выхода из типографии. Однако именно поэтому необходимо создавать фундаментальные руководства, которые не просто описывают текущие инструменты, а дают глубокое понимание принципов работы технологий. Эта книга – не попытка угнаться за трендами, а стремление зафиксировать базовые концепции и архитектурные решения, которые останутся актуальными независимо от смены версий библиотек и появления новых стартапов. Издание предназначено разработчикам, архитекторам систем, продуктовым менеджерам и всем, кто хочет не просто использовать готовые RAG-решения, но понимать, как они устроены изнутри. Каждая глава сочетает теоретические основы с практическими примерами, позволяя читателю не только изучить концепции, но и немедленно применить их на практике. В эпоху, когда ИИ-системы становятся критически важной инфраструктурой, понимание принципов их работы превращается из преимущества в необходимость.

Содержание

| | |
|---|-----------|
| Предисловие | 12 |
| Введение | 15 |
| Часть I. ОСНОВЫ RAG-ТЕХНОЛОГИЙ | 19 |
| Глава 1. Введение в RAG-системы | 20 |
| 1.1. Что такое RAG и зачем он нужен | 20 |
| Анатомия проблемы | 20 |
| Как RAG меняет правила игры | 21 |
| Практическая ценность для бизнеса | 21 |
| Ограничения и реалистичные ожидания | 22 |
| 1.2. Эволюция от поисковых систем к интеллектуальным системам | 22 |
| Эра лексического поиска: от простоты к совершенству | 22 |
| Границы лексического подхода | 23 |
| Революция векторных представлений | 23 |
| Гибридные решения: лучшее из двух миров | 23 |
| От поиска к пониманию: появление RAG | 24 |
| Интеллектуальные системы нового поколения | 24 |
| Будущее: от поиска к рассуждению | 24 |
| 1.3. Базовые принципы работы RAG | 25 |
| Двухмодульная архитектура: разделяй и властвуй | 25 |
| Этап подготовки: создание интеллектуального индекса | 25 |
| Векторный поиск: от слов к смыслам | 26 |
| Повторное ранжирование: уточнение релевантности | 26 |
| Генерация контекстуального ответа | 26 |
| Принцип непараметрической памяти | 27 |
| Многоэтапные и агентные RAG-системы | 27 |
| Адаптивность и контекстуальность | 27 |
| 1.4. Минимальный практический пример на Python | 28 |
| Установка зависимостей | 28 |
| Базовая реализация RAG | 28 |
| Демонстрация работы | 30 |
| Разбор ключевых моментов | 30 |
| Ограничения и возможности развития | 31 |
| Глава 2. Архитектура и компоненты | 32 |
| 2.1. Основные компоненты RAG-систем | 32 |
| Архитектурная схема RAG-системы | 32 |
| Конвейер данных: офлайн-подготовка | 33 |
| Обработка запросов: онлайн-поток | 33 |

| | |
|---|-----------|
| Формирование контекста и генерация | 34 |
| Дополнительные компоненты | 34 |
| Модульность и масштабируемость | 34 |
| 2.2. Архитектурные паттерны: от классики к автономным системам..... | 35 |
| Эволюция RAG-архитектур | 35 |
| Агентный RAG: автономность и адаптивность | 36 |
| Выбор архитектурного паттерна | 37 |
| 2.3. Интеграция компонентов интеллектуальной системы | 38 |
| Микросервисная архитектура RAG-систем | 38 |
| Паттерны интеграции: синхронность и асинхронность | 39 |
| Оркестрация процессов: дирижёр интеллектуальной системы | 40 |
| Стратегии управления данными | 40 |
| Обеспечение качества интеграции | 41 |
| Мониторинг и наблюдаемость | 41 |
| 2.4. Практический пример создания простейшей RAG-системы | 42 |
| Архитектура самодельной системы | 42 |
| Установка зависимостей | 43 |
| Ключевые особенности реализации..... | 47 |
| Глава 3. Векторные представления и эмбединги..... | 48 |
| 3.1. Принципы работы с векторными представлениями..... | 48 |
| Что такое векторные представления..... | 48 |
| Дистрибутивная гипотеза: основа всего..... | 49 |
| Семантическое пространство: география смыслов | 49 |
| Фундаментальные принципы работы с векторами..... | 50 |
| Практические следствия принципов | 50 |
| 3.2. Векторные базы данных и их применение | 51 |
| Принципы устройства векторного хранилища | 51 |
| Современные решения: ландшафт векторных баз данных..... | 51 |
| Критерии выбора: навигация в многообразии | 53 |
| Рекомендации по выбору..... | 53 |
| 3.3. Сегментирование текстов как основа RAG-сервисов | 53 |
| Фундаментальная проблема сегментирования..... | 54 |
| Метод фиксированного размера: простота как преимущество..... | 54 |
| Скользящее окно: сохранение контекста через перекрытие | 55 |
| Семантическое сегментирование: следование логике текста..... | 55 |
| Структурное разделение: использование архитектуры документа | 56 |
| Адаптивное сегментирование: искусственный интеллект в помощь ... | 57 |
| Критерии выбора метода сегментирования | 57 |
| 3.4. Код для работы с эмбедингами OpenAI и векторными БД..... | 58 |
| Базовая работа с эмбедингами OpenAI..... | 58 |
| Интеграция с ChromaDB: локальная векторная база данных | 60 |
| Работа с Qdrant: производительность и гибкость | 61 |
| Интеграция с Pinecone: облачная мощность | 64 |
| Комплексный пример: RAG-система с выбором векторной БД..... | 66 |

| | |
|---|-----|
| Часть II. ПРАКТИЧЕСКАЯ РЕАЛИЗАЦИЯ | 69 |
| Глава 4. Фреймворки и инструменты | 70 |
| 4.1. Обзор популярных фреймворков для RAG | 70 |
| Архитектура экосистемы RAG-фреймворков | 70 |
| Универсальные фреймворки: основа экосистемы | 71 |
| Специализированные решения: фокус на конкретные потребности | 72 |
| Корпоративные платформы: безопасность и интеграция | 72 |
| 4.2. Сравнительный анализ решений | 73 |
| 4.3. Выбор инструментов под задачу | 75 |
| Алгоритм принятия решений | 75 |
| Критерии для исследовательских проектов | 76 |
| Стратегии для прототипирования | 77 |
| Продакшн-системы: надёжность превыше всего | 77 |
| Корпоративные требования: безопасность и интеграция | 77 |
| Комплексные критерии оценки | 78 |
| Глава 5. Ретриверы и поиск информации | 79 |
| 5.1. Типы ретриверов и их особенности | 79 |
| Векторные ретриверы: семантическое понимание | 79 |
| Лексические ретриверы: точность терминологических совпадений | 80 |
| Гибридные ретриверы: синтез подходов | 80 |
| Кросс-энкодерные ретриверы: глубокое понимание | 81 |
| MMR: баланс релевантности и разнообразия | 81 |
| Специализированные ретриверы | 81 |
| 5.2. Алгоритм выбора и сравнительный анализ ретриверов | 82 |
| Первый этап: анализ типа запросов | 82 |
| Второй этап: приоритет семантического поиска | 83 |
| Третий этап: специальные требования | 83 |
| Критерии практического применения алгоритма | 84 |
| Сравнительный анализ ретриверов | 85 |
| 5.3. Детально о точных методах | 86 |
| TF-IDF: фундаментальная модель | 86 |
| Практическая реализация TF-IDF | 87 |
| BM25: вероятностная эволюция | 90 |
| Продвинутая реализация BM25 | 90 |
| Оптимизированная реализация для больших корпусов | 94 |
| 5.4. От точных методов до нейросетевых приближений | 97 |
| Архитектурная эволюция: от независимых терминов к контекстуальным представлениям | 97 |
| Практическая реализация гибридной системы поиска | 98 |
| Теоретические основы нейросетевого поиска | 105 |
| Практические компромиссы и выбор подхода | 105 |
| 5.5. Гибридный поиск и его преимущества | 105 |
| Концептуальные основы гибридизации | 106 |

| | |
|--|------------|
| Математические принципы комбинирования | 106 |
| Архитектурные преимущества | 107 |
| Качественные преимущества | 107 |
| Практические соображения внедрения | 107 |
| Эволюционные направления | 108 |
| Глава 6. Типизация и классификация RAG-сервисов | 109 |
| 6.1. Классификация RAG-систем по назначению | 109 |
| Разговорные RAG-системы | 110 |
| Аналитические RAG-системы | 111 |
| Контентные RAG-системы | 111 |
| Поисковые RAG-системы | 112 |
| Рекомендательные RAG-системы | 112 |
| Системы поддержки принятия решений | 113 |
| 6.2. Выбор подходящего типа для конкретной задачи | 113 |
| Алгоритм принятия решений | 114 |
| Первичная классификация по взаимодействию | 115 |
| Контентно-ориентированные решения | 115 |
| Поисковые архитектуры..... | 116 |
| Аналитические системы | 116 |
| Системы поддержки принятия решений | 116 |
| Критерии валидации выбора | 117 |
| 6.3. Сравнительный анализ подходов | 117 |
| Ключевые выводы сравнительного анализа | 119 |
| 6.4. Пример реализации рекомендательной системы..... | 119 |
| Практическая реализация рекомендательной системы | 120 |
| Часть III. ОПТИМИЗАЦИЯ И КОНТРОЛЬ КАЧЕСТВА | 127 |
| Глава 7. Оценка эффективности RAG-систем | 128 |
| 7.1. Метрики качества для RAG | 128 |
| Метрики качества поиска | 128 |
| Основные метрики фреймворка RAGAS | 129 |
| Дополнительные метрики оценки | 130 |
| Контекстуальные метрики..... | 130 |
| Метрики латентности и производительности..... | 130 |
| Холистические метрики | 131 |
| 7.2. Методы тестирования и валидации | 131 |
| Модульное тестирование компонентов RAG | 132 |
| Интеграционное тестирование взаимодействий..... | 132 |
| Комплексное тестирование пользовательских сценариев | 133 |
| А/В-тестирование и эксперименты..... | 133 |
| Специализированные методы валидации | 134 |
| 7.3. Инструменты для автоматической оценки..... | 134 |
| RAGAS – специализированный фреймворк для RAG..... | 135 |

| | |
|---|-----|
| TruLens – универсальная платформа наблюдения..... | 135 |
| DeepEval – комплексная система тестирования..... | 136 |
| LlamaIndex – встроенная оценка..... | 136 |
| Phoenix – мониторинг в реальном времени | 136 |
| Сравнительная таблица инструментов..... | 137 |
| Специализированные решения | 138 |
| 7.4. Код для метрик RAGAS и схожих систем оценки..... | 138 |
| Базовая архитектура системы оценки | 138 |

Глава 8. Проблемы точности и галлюцинаций..... 148

| | |
|---|-----|
| 8.1. Типичные проблемы RAG-систем | 148 |
| Проблемы поискового компонента..... | 148 |
| Проблемы качества данных..... | 149 |
| Проблемы стратегии сегментирования..... | 149 |
| Проблемы генеративного компонента | 150 |
| Архитектурные проблемы | 150 |
| Проблемы оценки и мониторинга..... | 150 |
| Проблемы предметной специфичности..... | 151 |
| 8.2. Методы борьбы с галлюцинациями | 151 |
| Архитектурные методы снижения галлюцинаций | 151 |
| Техники промпт-инженерии | 152 |
| Алгоритмические подходы..... | 152 |
| Методы обучения и дообучения | 153 |
| Постпроцессинговые методы..... | 153 |
| Интегрированные подходы | 154 |
| 8.3. Гарды и системы контроля | 154 |
| Входные гарды и валидация..... | 155 |
| Промежуточные системы контроля | 156 |
| Архитектура LLM-as-a-Judge | 156 |
| Выходные системы контроля | 157 |
| Специализированные архитектуры гардов | 157 |
| Мониторинг и обратная связь..... | 157 |
| Интеграционные аспекты..... | 158 |
| 8.4. Пример применения гардов..... | 158 |
| 8.5. Практические решения для повышения достоверности | 169 |
| Система множественной верификации..... | 169 |
| Практические рекомендации..... | 176 |

Глава 9. Память и контекст..... 177

| | |
|--|-----|
| 9.1. Подсистемы памяти для ИИ-агентов..... | 177 |
| Краткосрочная память и рабочий контекст | 178 |
| Эпизодическая память: хранение опыта | 179 |
| Семантическая память: структурированные знания | 179 |
| Процедурная память: навыки и алгоритмы | 179 |
| Эмоциональная память: персонализация и социальный контекст | 180 |

| | | |
|--|---|------------|
| | Ассоциативно-гетерархическая память: объединение всех видов памяти..... | 180 |
| | Интеграция с RAG-системами..... | 182 |
| 9.2. | Управление долгосрочным и краткосрочным контекстами | 182 |
| | Архитектура управления контекстом | 183 |
| | Краткосрочное управление контекстом | 183 |
| | Долгосрочное управление контекстом | 184 |
| | Интеграция с RAG-системами..... | 184 |
| | Адаптивные стратегии управления | 184 |
| | Технические реализации и оптимизации | 185 |
| 9.3. | Персонализация через память..... | 185 |
| | Механизмы персональной адаптации | 186 |
| | Уровни персонализации..... | 186 |
| | Динамическое обучение предпочтений | 186 |
| | Техническая реализация | 187 |
| | Этические аспекты и приватность | 187 |
| 9.4. | Реализация подсистемы памяти..... | 188 |
| Часть IV. ОТРАСЛЕВОЕ ПРИМЕНЕНИЕ | | 201 |
| Глава 10. Корпоративные решения | | 202 |
| 10.1. | RAG в управлении знаниями предприятия..... | 202 |
| | Архитектура корпоративных RAG-систем | 202 |
| | Трансформация корпоративных процессов | 203 |
| | Специализированные корпоративные применения | 203 |
| | Измеримые результаты внедрения..... | 204 |
| | Стратегические преимущества | 204 |
| | Вызовы и ограничения корпоративного внедрения | 204 |
| 10.2. | Интеграция с существующими системами | 205 |
| | Архитектурные паттерны интеграции | 205 |
| | Интеграция с ERP-системами | 206 |
| | Интеграция с CRM-системами | 206 |
| | Интеграция с системами документооборота и базами знаний..... | 206 |
| | Облачные хранилища и файловые системы | 207 |
| | Системы управления проектами | 207 |
| | Техническая реализация интеграций | 207 |
| | Безопасность интеграций..... | 208 |
| | Перспективы развития интеграций..... | 208 |
| 10.3. | Безопасность и конфиденциальность данных | 209 |
| | Специфические угрозы RAG-систем | 209 |
| | Архитектурные решения для обеспечения безопасности | 209 |
| | Дифференциальная приватность в RAG | 210 |
| | Управление доступом и авторизация | 210 |
| | Техническая защита векторных данных | 211 |
| | Соответствие регуляторным требованиям | 211 |
| | Мониторинг и обнаружение угроз..... | 211 |

| | |
|--|-----|
| Перспективы развития безопасности | 212 |
| 10.4. Готовые решения для корпораций | 212 |
| Облачные RAG-платформы | 212 |
| Специализированные корпоративные решения | 213 |
| Отраслевые решения | 213 |
| Готовые фреймворки и инструменты | 214 |
| Российские решения и локализация | 214 |
| Критерии выбора готовых решений | 214 |
| Тенденции развития рынка | 215 |
| Глава 11. Специализированные применения | 216 |
| 11.1. RAG в юридической сфере | 216 |
| 11.2. Медицинские приложения | 218 |
| 11.3. Образовательные технологии | 221 |
| Глава 12. Интеллектуальная поддержка клиентов | 224 |
| 12.1. Поддержка пользователей на основе RAG | 224 |
| 12.2. Чат-боты нового поколения | 225 |
| 12.3. Интеграция с CRM и другими системами | 227 |
| 12.4. Полный код чат-бота с RAG | 228 |
| Основной код чат-бота | 228 |
| Файл зависимостей requirements.txt | 236 |
| Пример файла настроек .env | 237 |
| Инструкции по запуску | 237 |
| Часть V. ПРОДВИНУТЫЕ ТЕХНИКИ | 239 |
| Глава 13. RAG vs Fine-tuning | 240 |
| 13.1. Сравнительный анализ подходов | 240 |
| Retrieval-Augmented Generation | 240 |
| Full Fine-Tuning | 240 |
| Low-Rank Adaptation (LoRA) | 241 |
| Quantized Low-Rank Adaptation (QLoRA) | 242 |
| Prompt Tuning | 242 |
| Prefix Tuning | 242 |
| Адаптеры | 243 |
| Сравнительная таблица подходов | 243 |
| 13.2. Критерии выбора стратегии | 245 |
| 13.3. Гибридные решения | 246 |
| Архитектурные паттерны гибридных систем | 246 |
| RAG-Tuned-LLM: объединение принципов | 247 |
| Hybrid RAG: многоканальный поиск | 247 |
| REFINE: совместная оптимизация компонентов | 248 |
| SmartRAG: совместное обучение задач | 248 |

| | |
|--|------------|
| Практические сценарии применения | 248 |
| Компромиссы и вызовы | 249 |
| Глава 14. Мультимодальные RAG-системы | 250 |
| 14.1. Работа с изображениями и документами..... | 250 |
| Архитектурные подходы к мультимодальному RAG | 250 |
| Обработка визуально насыщенных документов..... | 251 |
| Подготовка и предобработка визуальных данных..... | 251 |
| Практические реализации и инструменты | 252 |
| Бенчмарки и оценка качества | 252 |
| 14.2. Обработка видео- и аудиоконтента..... | 253 |
| Архитектура VideoRAG | 253 |
| Обработка аудиоконтента..... | 254 |
| Multi-RAG: унификация мультимодальной информации | 254 |
| SceneRAG: сегментация на уровне сцен | 255 |
| Практическая реализация мультимодального аудио/видео RAG | 255 |
| Применение в различных предметных областях..... | 256 |
| 14.3. Интеграция различных типов данных..... | 256 |
| Низкоуровневое слияние: унифицированное векторное пространство..... | 257 |
| Высокоуровневое слияние: отдельная обработка и объединение..... | 257 |
| Гибридное слияние: комбинирование подходов | 258 |
| Кросс-модальное выравнивание и проекция | 258 |
| Практическая реализация интеграции..... | 259 |
| Инструменты и платформы..... | 260 |
| Вызовы и решения | 261 |
| 14.4. Пример мультимодального пайплайна | 261 |
| Код мультимодального RAG-пайплайна..... | 262 |
| Описание реализации | 267 |
| Глава 15. Масштабирование и производительность | 269 |
| 15.1. Оптимизация производительности RAG..... | 269 |
| 15.2. Горизонтальное и вертикальное масштабирования..... | 271 |
| 15.3. Кеширование и оптимизация запросов..... | 272 |
| 15.4. Архитектурные решения для высоконагруженных систем | 273 |
| Заключение | 275 |
| Приложение А. Глоссарий терминов..... | 277 |
| Приложение Б. Ресурсы для дальнейшего изучения | 280 |
| Приложение В. Об истинной интеллектуальной системе | 283 |

Предисловие

Дорогие читатели!

Более тридцати лет я занимаюсь искусственным интеллектом. За эти годы я воочию видел, как наша отрасль переживала и зимы, и весны, как горделивые надежды сменялись разочарованием, а скепсис – новыми прорывами. Но последние несколько лет стали особенными. Мы стали свидетелями появления технологий, которые впервые приблизили нас к мечте о настоящем искусственном интеллекте.

RAG-системы – не столько очередная аббревиатура в бесконечной очереди терминов, но фундаментальный сдвиг в том, как мы представляем взаимодействие между машинами и человеческими знаниями. За последний год я написал более 40 статей и заметок о различных аспектах RAG-технологий, опубликованных в телеграм-канале «Технооптимисты» (https://t.me/drv_official). Каждая из них была попыткой зафиксировать момент истины в стремительно развивающейся области.

Когда издательство «ДМК Пресс» предложило систематизировать эти знания в книге, я понял, что стою перед классической дилеммой технического писателя: как написать книгу о технологии, которая развивается быстрее, чем пишется текст. Каждую неделю появляются новые фреймворки, подходы, архитектурные решения. К моменту выхода книги многие конкретные инструменты и версии библиотек неизбежно устареют.

Но именно поэтому эта книга нужна. В эпоху информационного шума критически важно не просто следить за новинками, но понимать фундаментальные принципы. RAG-системы – это не мода, это новая парадигма работы с информацией. И как любая парадигма, она имеет базовые законы, архитектурные принципы и философию, которые останутся актуальными независимо от смены конкретных инструментов.

Я видел, как компании тратят месяцы на внедрение RAG-решений, не понимая базовых принципов их работы. Как разработчики пытаются «прикрутить» векторный поиск к существующим системам, получая в результате дорогие и неэффективные решения. Как менеджеры продуктов ожидают от RAG-систем магии, а получают очередной источник галлюцинаций и технического долга.

Цель этой книги – дать читателю именно то понимание, которого так часто не хватает на практике. Не просто «как запустить библиотеку X с параметрами Y», а «почему эта архитектура работает именно так», «какие компромиссы скрываются за каждым решением», «как оценить качество системы до того, как вложить в неё серьёзные ресурсы».

Я пытался сохранить баланс между теоретической глубиной и практической применимостью. В каждой главе есть примеры кода, но они служат ил-

люстрацией концепций, а не самоцелью. Главная задача – научить читателя мыслить категориями RAG-архитектур, понимать их возможности и ограничения, принимать обоснованные технические решения.

Особое внимание в книге уделено отраслевым применениям RAG-систем. За эти годы я консультировал компании от стартапов до крупных корпораций, работал с проектами в юриспруденции, медицине, образовании, финансах. В каждой отрасли RAG решает свои уникальные задачи, имеет специфические требования и ограничения. Понимание этого контекста критически важно для успешного внедрения технологий.

Я благодарен всем своим коллегам и студентам, читателям канала «Техно-оптимисты» и слушателям одноимённого подкаста, чьи вопросы и комментарии помогали глубже понять нюансы RAG-технологий. Особая благодарность команде издательства «ДМК Пресс» за терпение и профессионализм в работе с материалом, который буквально менялся по ходу написания под влиянием новых технологических решений.

RAG-системы – это мост между традиционными подходами к поиску информации и будущими интеллектуальными помощниками. Построение этого моста требует понимания как классических алгоритмов информационного поиска, так и современных достижений в области больших языковых моделей. Надеюсь, эта книга поможет вам не просто освоить RAG-технологии, но и стать их осознанными архитекторами и пользователями.

Интересно, что корни моего интереса к проблемам создания истинных интеллектуальных систем уходят в далекие девяностые. Ещё будучи студентом МИФИ, на пороге получения диплома, я вместе с моим товарищем Владимиром Юрьевичем Степаньковым написал дерзкую работу «Об истинной интеллектуальной системе». В ней мы, молодые и амбициозные, критиковали современные нам подходы к искусственному интеллекту за фундаментальную проблему GIGO (garbage in – garbage out) – неспособность систем генерировать новые знания сверх заложенных разработчиками.

Мы отправили эту работу на студенческую конференцию, не согласовав с научным руководителем – поступок, который чуть не стоил нам дипломов – нас грозились отчислить из университета «с волчьим билетом». Но именно в той неопубликованной и потерянной на почти 30 лет¹ работе были

¹ Мы написали статью «Об истинной интеллектуальной системе» в 1998 году и отправили её на студенческую конференцию, которая проходила в Судаке, а после отказа оргкомитета и жёсткой выволочки со стороны руководства кафедры мы забыли про неё (но я-то помнил всегда). Временами за чашечкой душистого чая я напоминал Владимиру Юрьевичу о том, что когда-то мы написали такую статью, но он отрицал сам факт этого – вероятно, настолько его поразил получившийся эффект, что мозг решил компенсировать это буквальной выборочной амнезией. И вот в 2023 году во время празднования 60-летия кафедры № 22 «Кибернетика» НИЯУ МИФИ мы с ним нашли оригинальный текст этой статьи на одном из старых компьютеров учебно-научной лаборатории «Системы искусственного интеллекта» кафедры. Полный текст этой статьи с минимальными правками, касающимися исправления пары орфографических ошибок и типографики, приведён в приложении Г.

сформулированы требования к интеллектуальным системам, которые сегодня обретают новую актуальность в контексте RAG-технологий. Мы писали о необходимости самостоятельного целеполагания, автоматической генерации новых знаний, взаимодействии с человеческим социумом – всё то, что сегодня становится реальностью в современных RAG-системах.

Добро пожаловать в увлекательное время развития поистине интеллектуальных систем!

Роман Викторович Душкин

Эксперт в области ИИ

Генеральный директор ООО «А-Я эксперт»

Руководитель образовательной программы

«Искусственный интеллект» НИЯУ МИФИ

Серпухов – Москва, октябрь 2025 года

Введение

Мы живём в эпоху интеллектуального переворота. За последние три года произошло нечто, что изменило наше понимание возможностей искусственного интеллекта – от экспериментальных прототипов мы перешли к системам, способным рассуждать, анализировать и создавать контент на уровне, который ещё недавно казался недостижимым. В центре этой революции находится технология RAG (Retrieval-Augmented Generation) – подход, который превратил большие языковые модели из «умных стохастических попугаев» в действительно интеллектуальные системы.

Цифры говорят сами за себя: доля приложений с RAG выросла с 31 % до 51 % всего за один 2024 год. Более половины компаний теперь используют RAG-технологии, тогда как доля дорогостоящего файн-тюнинга упала до 9 %. Венчурные инвестиции в стартапы, специализирующиеся на генеративном искусственном интеллекте, в третьем квартале 2024 года достигли рекордных 3,9 млрд дол. – на 65 % больше, чем годом ранее. В России объём проектов по работе с большими данными и искусственным интеллектом вырос на 40 %, и среди наиболее востребованных технологий эксперты называют именно RAG.

Почему именно сейчас?

RAG стал катализатором массового внедрения корпоративных систем искусственного интеллекта по простой причине: он решает фундаментальную проблему доступа к знаниям. Традиционные языковые модели, какими бы впечатляющими ни были их возможности, ограничены данными, на которых они обучены. Они не знают о событиях, произошедших после их обучения, не имеют доступа к корпоративной информации, внутренним документам, специализированным базам знаний.

RAG кардинально изменил эту ситуацию, позволив языковым моделям динамически обращаться к актуальным внешним источникам информации. Вместо попыток «втиснуть» все знания мира в параметры модели система получила возможность искать нужную информацию в момент формирования ответа, комбинируя мощь современных моделей с гибкостью поисковых систем.

Масштаб трансформации

Влияние RAG-технологий выходит далеко за рамки технических улучшений. Это новая парадигма работы с корпоративными знаниями. В технической поддержке RAG снижает количество эскалаций на 30–40 %, а время обработки стандартных заявок сокращается на 50–60 %. В юридических и аналити-

ческих подразделениях системы высвобождают до 8 часов рабочего времени еженедельно, автоматизируя поиск и анализ документов.

Более того, RAG становится основой для следующего поколения ИИ-агентов – систем, способных не просто отвечать на вопросы, но выполнять сложные многоэтапные задачи, обращаясь к различным источникам данных и принимая обоснованные решения. По текущим экспертным прогнозам, к 2026 году более 80 % корпоративных ИИ-проектов будут использовать гибридные архитектуры, включающие большие языковые модели и методы доступа к внешним данным.

Вызовы и реальность внедрения

Однако путь к успешному внедрению RAG-систем не так прост, как может показаться из маркетинговых материалов. Качество RAG-системы напрямую зависит от качества данных, с которыми она работает. Дублирующиеся документы, устаревшая информация, противоречивые данные – все эти проблемы усиливаются в RAG-системах и могут привести к генерации некорректных ответов.

Исследования показывают, что RAG может значительно увеличить вероятность получения опасных или некорректных ответов, особенно при работе с неструктурированными или плохо организованными данными. Это подчёркивает критическую важность правильной архитектуры системы и глубокого понимания принципов её работы.

От хайпа к реальной пользе

За несколько лет работы с RAG-технологиями я видел, как компании переходят от первоначального энтузиазма к более взвешенному пониманию возможностей и ограничений этих систем. Успешные внедрения объединяет несколько общих черт: чёткое понимание бизнес-задач, тщательная подготовка данных, правильный выбор архитектурных решений и непрерывный мониторинг качества системы.

RAG – не универсальное решение для всех задач корпоративной ИИ-системы. Для генерации креативного контента лучше использовать классические языковые модели без поискового компонента. Для задач, требующих глубокого понимания узкой предметной области, может потребоваться специализированное дообучение модели. RAG наиболее эффективен там, где нужно сочетать общие языковые способности с доступом к специфическим, часто обновляющимся знаниям.

Структура и цели книги

Эта книга построена как практическое путешествие от базовых концепций к реальным корпоративным внедрениям. Первая часть знакомит читателя с фундаментальными принципами RAG-систем, их архитектурой и ключевыми компонентами, а вторая часть погружает его в технические детали: от выбора правильных фреймворков до настройки компонентов поиска и генерации.

Третья часть посвящена критически важным вопросам качества и надёжности – как оценить эффективность системы, как бороться с галлюцинациями, как обеспечить стабильную работу в производственной среде. Четвёртая часть исследует отраслевые применения RAG: от корпоративных систем управления знаниями до специализированных решений в медицине, юриспруденции, образовании.

Заключительная часть рассматривает перспективные направления развития: мультимодальные RAG-системы, интеграцию с агентными архитектурами, новые подходы к масштабированию и оптимизации. Каждая глава сочетает теоретические основы с практическими примерами, позволяя читателю не только понять концепции, но и немедленно применить их на практике.

Для кого написана эта книга

Эта книга написана для практиков, а не теоретиков. Она адресована разработчикам, которые хотят создавать эффективные RAG-системы, архитекторам, проектирующим корпоративные ИИ-платформы, продуктовым менеджерам, принимающим решения о внедрении ИИ-технологий, и руководителям, которым нужно понимать возможности и ограничения современных интеллектуальных систем.

Предполагается, что читатель имеет базовые знания в области программирования и машинного обучения, но при этом не требуется глубокая экспертиза в области обработки естественного языка или теории информационного поиска. Главная цель – дать практическое понимание того, как RAG-системы работают, почему они работают именно так и как их можно эффективно использовать для решения реальных бизнес-задач.

Методология подачи материала

В отличие от чисто академических трудов или поверхностных обзоров, эта книга следует принципу «от проблемы к решению». Каждая глава начинается с реальной практической задачи, затем объясняет теоретические основы её решения и завершается конкретными примерами реализации. Такой подход позволяет читателю не просто изучить инструменты, но понять логику их применения.

Особое внимание уделяется архитектурным принципам и паттернам проектирования. RAG-системы – это не просто набор библиотек и API, а сложные информационные системы, требующие продуманного подхода к проектированию, тестированию и эксплуатации. Понимание этих принципов критически важно для создания надёжных и масштабируемых решений.

Актуальность и перспективы

RAG-технологии развиваются с головокружительной скоростью. Многообразие инструментов для повышения точности RAG-систем стремительно растёт: от продвинутых техник переранжирования до гибридных архитектур,

объединяющих несколько типов поиска. В январе 2025 года появились исследования, расширяющие концепцию RAG на видеоконтент, что открывает новые возможности для мультимодальных моделей.

Однако в погоне за технологическим прогрессом важно не потерять понимание фундаментальных принципов. Независимо от того, какие новые фреймворки и подходы появятся в ближайшие годы, базовые концепции поиска релевантной информации, оценки её качества и включения в процесс генерации останутся актуальными. Именно эти принципы и составляют основу данной книги.

RAG – это не просто техническое решение, это новый способ мышления об искусственном интеллекте как о системе, способной учиться, адаптироваться и расти вместе с накопленными знаниями. В эпоху, когда объём информации растёт экспоненциально, а скорость принятия решений становится критически важным конкурентным преимуществом, понимание и эффективное применение RAG-технологий превращается из полезного навыка в стратегическую необходимость.

ЧАСТЬ I



Основы RAG-технологий

Глава 1

Введение в RAG-системы

1.1. Что такое RAG и зачем он нужен

Современному состоянию развития больших языковых моделей присущ странный парадокс, который можно даже назвать фундаментальной проблемой: несмотря на впечатляющие возможности GPT-5, Claude 4 или других передовых решений, их знания ограничены данными, на которых они были обучены. Представьте, что у вас есть блестящий сотрудник, который знает практически всё о мире, но его знания «заморожены» на определённой дате и более не обновляются.

RAG (Retrieval-Augmented Generation) – *дополненная поиском генерация* – архитектурный подход, который решает эту проблему, соединяя большую языковую модель с внешней, постоянно обновляемой базой знаний. Вместо того чтобы полагаться исключительно на «вшитые» в модель знания, RAG-система сначала ищет релевантную информацию в специализированных источниках данных, а затем использует найденные факты для формирования ответа.

Анатомия проблемы

Традиционные большие языковые модели страдают от нескольких критических ограничений. *Галлюцинации*¹ – пожалуй, самая известная проблема, когда модель с удивительной уверенностью сообщает заведомо ложную информацию. Это происходит потому, что модель пытается дать ответ на любой вопрос, даже если у неё нет достоверных данных по теме.

¹ Более корректное название феномена – *конфабуляция*, то есть такое расстройство памяти, которое нередко сопровождает прогрессирующую амнезию: события, действительно имевшие место, забываются, а возникающие пробелы в памяти восполняются вымыслами.

Второй критический недостаток – устаревание данных. Если модель обучена на данных до 2023 года, она не знает о событиях 2024-го и будет давать устаревшие ответы на вопросы о текущих событиях. Для корпоративного использования это особенно болезненно: новые продукты, изменения в политиках компании, актуальные цены – всё это остаётся за пределами знаний модели.

Третья проблема – отсутствие специализированных знаний. Общие модели хорошо справляются с популярными темами, но могут давать поверхностные или неточные ответы по узкоспециализированным вопросам: внутренним процессам компании, отраслевым регламентам, техническим спецификациям оборудования.

Как RAG меняет правила игры

RAG кардинально меняет подход к решению этих проблем. Вместо попыток «запихнуть» все знания мира в параметры модели система получает возможность динамически обращаться к внешним источникам в момент формирования ответа.

Процесс работает следующим образом: когда поступает запрос пользователя, RAG-система сначала преобразует его в векторное представление и ищет наиболее релевантные фрагменты в предварительно проиндексированной базе знаний. Найденная информация затем передаётся языковой модели как дополнительный контекст, на основе которого формируется итоговый ответ.

Ключевое преимущество такого подхода – разделение хранения знаний и их обработки. Мы можем обновлять базу знаний в режиме реального времени, добавляя новые документы, корректируя устаревшие данные, интегрируя отраслевую специфику – и всё это без необходимости перетренировки большой языковой модели.

Практическая ценность для бизнеса

Для корпоративных информационных систем RAG открывает принципиально новые возможности. Корпоративный помощник на базе RAG может мгновенно находить информацию в многотысячных массивах внутренних документов: от политик до технических спецификаций продуктов. Система клиентской поддержки получает доступ к актуальной базе знаний о продуктах и услугах, предоставляя точные ответы без риска передачи устаревшей информации.

Особенно ценно то, что RAG позволяет создавать узкоспециализированные интеллектуальные системы без астрономических затрат на обучение собственных моделей. Вместо этого можно взять готовую большую (или даже малую) языковую модель и «научить» её работать с конкретной предметной областью через настройку компонента поиска.

Ограничения и реалистичные ожидания

Несмотря на впечатляющие возможности, RAG – не панацея от всех проблем языковых моделей. Качество ответов напрямую зависит от качества данных в базе знаний. Если исходные документы содержат противоречивую или неточную информацию, система может усиливать эти проблемы.

RAG также не полностью устраняет галлюцинации – модель всё ещё может творчески интерпретировать найденную информацию или заполнять пробелы собственными «додумываниями». Именно поэтому современные RAG-системы дополняются механизмами контроля качества и проверки фактов.

Тем не менее правильно построенная RAG-система кардинально улучшает надёжность и актуальность ответов языковых моделей, превращая их из «говорящих энциклопедий прошлого» в динамические интеллектуальные системы, способные работать с живой, постоянно обновляющейся информацией.

1.2. Эволюция от поисковых систем к интеллектуальным системам

История поиска информации – это история постоянной борьбы с фундаментальной проблемой понимания смысла. От простейших каталогов библиотек до современных RAG-систем пройден долгий путь технологических революций, каждая из которых приближала нас к идеалу: системе, способной не просто найти документы по ключевым словам, но понять, что именно ищет пользователь, и дать осмысленный ответ.

Эра лексического поиска: от простоты к совершенству

Первое поколение поисковых технологий базировалось на принципе прямого соответствия слов. Подход TF-IDF (Term Frequency-Inverse Document Frequency), появившийся в 1970-х годах, стал фундаментом информационного поиска на десятилетия вперёд. Его гениальность заключалась в простоте: часто встречающиеся в документе слова получают высокий вес, а редкие в коллекции термины считаются более значимыми.

Однако TF-IDF имел критический недостаток – проблему масштабирования по длине документов. Два документа разной длины с одинаковым содержанием получали различные оценки релевантности, что искажало результаты поиска. Именно для решения этой проблемы был разработан

алгоритм BM25 (Best Matching 25), который стал золотым стандартом лексического поиска.

Алгоритм BM25 ввёл концепцию насыщения частоты терминов – идею о том, что увеличение количества вхождений слова в документ даёт убывающую отдачу. Если в статье о Ломоносове его имя упоминается 12 раз вместо шести, это не делает статью в два раза более релевантной – закон убывающей полезности действует и в информационном поиске.

Границы лексического подхода

Несмотря на техническое совершенство, лексические методы столкнулись с непреодолимым барьером *семантического разрыва*. Система, основанная на точном совпадении слов, не могла понять, что «автомобиль» и «машина» – синонимы, а «покупка автомобиля» и «приобретение транспортного средства» описывают одно и то же действие.

Попытки решения через расширение запросов синонимами и создание тезаурусов оказались громоздкими и неэффективными. Язык развивается быстрее, чем можно обновлять словари, а контекстные значения слов делают автоматическое расширение запросов источником шума. Стало очевидно: для прорыва в качестве поиска нужен принципиально иной подход.

Революция векторных представлений

Семантическая революция началась с появлением искусственных нейронных сетей, способных преобразовывать слова в числовые векторы – *эмбединги*. Технологии Word2Vec, а позднее BERT и их наследники позволили кодировать смысловое содержание текста в многомерном пространстве, в котором семантически близкие понятия располагаются рядом.

Векторный поиск кардинально изменил правила игры. Вместо поиска точных совпадений система начала искать семантическое сходство между запросом и документами. Запрос «найти информацию о покупке автомобиля» мог найти документы со словами «приобретение», «машина», «транспорт» – поисковая система научилась понимать смысл, а не только буквы.

Ключевым преимуществом стала способность поисковых систем работать с запросами на разговорном языке. Пользователь мог задать вопрос «Как лучше выбрать подержанную машину?» и получить релевантные результаты, даже если в найденных документах не было точного совпадения фразы.

Гибридные решения: лучшее из двух миров

Практическое применение показало, что идеальный поиск требует сочетания лексических и семантических подходов. Векторный поиск превосходно

работает с концептуальными и абстрактными запросами, но может «терять» точные совпадения имён, артикулов, специальных терминов.

Современные системы используют гибридный подход, комбинируя алгоритм BM25 для точных лексических совпадений с векторным поиском для семантического понимания. Такая архитектура обеспечивает как точность в поиске конкретной информации, так и гибкость в понимании пользовательских интенций.

От поиска к пониманию: появление RAG

Следующий эволюционный скачок произошёл с появлением больших языковых моделей и концепции RAG. Если традиционные поисковые системы могли только найти и ранжировать документы, то RAG-системы способны понять найденную информацию и синтезировать осмысленный ответ.

RAG объединил три ключевых компонента: продвинутый поисковый движок (часто гибридный), базу знаний с возможностью динамического обновления и генеративную языковую модель, способную создавать связные ответы на основе найденной информации. Это позволило перейти от парадигмы «найти документы» к парадигме «получить ответ».

Интеллектуальные системы нового поколения

Современные RAG-системы представляют собой полноценные интеллектуальные платформы, способные не только искать информацию, но и рассуждать, анализировать противоречия, учитывать контекст предыдущих взаимодействий. Они могут работать с мультимодальным контентом – текстами, изображениями, структурированными данными, – предоставляя пользователю единый интерфейс для работы с разнородной информацией.

Ключевое отличие современных RAG-систем от классических поисковых машин заключается в способности к диалоговому взаимодействию. Система не только находит ответ на прямой вопрос, но может уточнить неясные моменты, предложить альтернативные интерпретации запроса, объяснить свою логику.

Будущее: от поиска к рассуждению

Эволюция продолжается в направлении создания систем, способных к глубокому рассуждению и обнаружению новых знаний. Следующее поколение RAG-систем будет не просто искать и пересказывать существующую информацию, но анализировать её, выявлять закономерности, делать выводы и генерировать новые гипотезы.

Путь от простого поиска по ключевым словам до интеллектуальных систем, способных рассуждать и творить¹, занял более полувека. RAG представляет собой не конечную точку этого пути, а важную веху в создании интеллектуальных систем, которые смогут стать настоящими партнёрами человека в работе с информацией и знаниями.

1.3. Базовые принципы работы RAG

Понять RAG – значит разобраться в элегантной простоте его архитектурного решения. В основе любой RAG-системы лежит фундаментальная идея разделения задач: поиск релевантной информации и её интеллектуальная обработка выполняются разными, специализированными компонентами. Эта архитектурная философия позволила преодолеть ключевые ограничения традиционных языковых моделей и создать системы, способные работать с динамически обновляемыми знаниями.

Двухмодульная архитектура: разделяй и властвуй

RAG-система состоит из двух ключевых компонентов: модуля поиска (*retriever*) и модуля генерации (*generator*), работающих в строгой последовательности. Эта архитектура решает сразу несколько проблем: специализация позволяет оптимизировать каждый компонент под свою задачу, а разделение обеспечивает гибкость и возможность независимого улучшения каждой части системы.

Модуль поиска специализируется на понимании смысла запроса и нахождении релевантных фрагментов информации в базе знаний. Его задача – не просто найти документы с совпадающими ключевыми словами, а идентифицировать контент, семантически связанный с запросом пользователя. Модуль генерации, в свою очередь, берёт найденную информацию и синтезирует связный ответ, адаптированный под конкретный контекст запроса.

Этап подготовки: создание интеллектуального индекса

Прежде чем RAG-система сможет отвечать на запросы, необходимо подготовить базу знаний в форме, оптимизированной для семантического поиска. Этот процесс включает несколько критически важных шагов, каждый из которых влияет на качество итоговой системы.

¹ В той мере, насколько понятия «рассуждение» и «творчество» применимы к ИИ на текущем этапе развития. – *Прим. ред.*

Сегментирование (разбивка документов) – первый и один из самых важных этапов подготовки данных. Длинные документы разбиваются на логически связанные фрагменты – *чанки* (chunk), каждый из которых должен содержать завершённую мысль или концепцию. Размер чанков критически важен: слишком маленькие не содержат достаточного контекста, слишком большие могут не поместиться в окно внимания языковой модели.

Векторизация превращает текстовые фрагменты в числовые представления – эмбединги. Модели эмбединга, такие как BERT или более современные решения, анализируют семантическое содержание текста и кодируют его в многомерный вектор. Ключевое свойство эмбедингов – семантически близкие тексты получают похожие векторные представления, что делает возможным поиск по смыслу, а не только по точным совпадениям слов.

Векторный поиск: от слов к смыслам

Когда пользователь задаёт вопрос, тот же процесс векторизации применяется и к запросу. Полученный вектор запроса сравнивается с векторами всех чанков в базе знаний с помощью метрик сходства – чаще всего косинусного расстояния. Система находит K наиболее похожих фрагментов (обычно от 3 до 10), которые считаются наиболее релевантными для ответа на вопрос.

Критически важно понимать: векторный поиск работает с семантическим сходством, а не с лексическим совпадением. Запрос «Как починить автомобиль?» может найти документы со словами «ремонт», «машина», «техническое обслуживание» – система понимает связь между концепциями, даже если точные слова не совпадают.

Повторное ранжирование: уточнение релевантности

Современные RAG-системы часто включают дополнительный этап *переранжирования* (reranking) – повторного ранжирования найденных фрагментов. Специализированные модели-ранкеры анализируют найденные чанки в контексте конкретного запроса и могут изменить их порядок, более точно оценив релевантность. Это особенно важно, когда векторный поиск находит семантически близкие, но не совсем подходящие фрагменты.

Генерация контекстуального ответа

На финальном этапе найденные фрагменты объединяются с исходным запросом и передаются языковой модели. Модель получает не просто вопрос пользователя, а расширенный контекст, включающий релевантную информацию из базы знаний. Примерный промпт выглядит так:

Контекст: [найденные фрагменты]

Вопрос: [запрос пользователя]

Ответ на вопрос, основываясь на предоставленном контексте.

Языковая модель использует как свои параметрические знания, так и предоставленный контекст для формирования ответа. Это решает проблему устаревания знаний – модель может опираться на актуальную информацию из внешних источников, даже если эти данные появились после её обучения.

Принцип непараметрической памяти

RAG фактически превращает статичную языковую модель в систему с динамической, обновляемой памятью. Знания больше не «зашиты» в параметрах модели, а хранятся во внешней базе знаний, которую можно обновлять, расширять и модифицировать без перетренировки модели. Это создаёт непараметрическую память – знания, существующие независимо от весов нейронной сети.

Многоэтапные и агентные RAG-системы

Современные реализации RAG выходят за рамки простой схемы «один запрос – один поиск – один ответ». Системы с цепочкой поиска могут выполнять несколько итераций поиска, уточняя запросы на основе промежуточных результатов. Агентные RAG-архитектуры используют языковую модель для принятия решений о том, какие функции поиска вызывать и как обрабатывать результаты.

Некоторые системы реализуют иерархический поиск: сначала грубый поиск по большим фрагментам, затем точный поиск внутри выбранных разделов. Это позволяет балансировать между эффективностью (скоростью поиска) и точностью (качеством найденной информации).

Адаптивность и контекстуальность

Ключевое преимущество RAG заключается в способности адаптироваться к контексту запроса. В отличие от статичных баз знаний, которые предоставляют фиксированные ответы на фиксированные вопросы, RAG-система может находить различные аспекты информации в зависимости от того, как сформулирован запрос. Один и тот же документ может быть релевантен для разных вопросов, но RAG будет извлекать из него разную информацию в зависимости от контекста.

Несмотря на появление новых техник и оптимизаций, базовые принципы RAG остаются неизменными. Понимание этих принципов – разделения поиска и генерации, векторного представления знаний, семантического поиска

и контекстуальной генерации – даёт прочную основу для создания эффективных интеллектуальных систем в любой предметной области.

1.4. Минимальный практический пример на Python

Лучший способ понять RAG – увидеть, как он работает на практике. Создадим простейшую RAG-систему, которая демонстрирует все ключевые принципы: векторизацию документов, семантический поиск и генерацию ответов на основе найденного контекста. Чтобы наша система смогла отвечать на вопросы по набору документов, достаточно всего несколько десятков строк кода.

Установка зависимостей

Для начала установим необходимые библиотеки:

```
pip install sentence-transformers faiss-cpu openai
```

Библиотека `sentence-transformers` будет генерировать эмбединги, FAISS послужит векторной базой данных, а OpenAI API – языковой моделью для генерации ответов¹.

Базовая реализация RAG

```
import numpy as np
import faiss
from sentence_transformers import SentenceTransformer
import openai
from typing import List

class SimpleRAG:
    def __init__(self, api_key: str):
        # Инициализируем модель для создания эмбедингов
        self.encoder = SentenceTransformer('all-MiniLM-L6-v2')

        # Настраиваем OpenAI
        openai.api_key = api_key

        # Хранилище для документов и индекс FAISS
        self.documents = []
        self.index = None
```

¹ Важно отметить, что для запуска примеров с использованием API ведущих языковых моделей, типа Open AI или Anthropic, здесь и далее необходимо обеспечить беспрепятственный доступ к этим сервисам.

```

def add_documents(self, documents: List[str]):
    """Добавляем документы в базу знаний"""
    self.documents.extend(documents)

    # Генерируем эмбединги для всех документов
    embeddings = self.encoder.encode(self.documents)

    # Создаём FAISS индекс для быстрого поиска
    dimension = embeddings.shape[1]
    self.index = faiss.IndexFlatIP(dimension) # Inner Product для косинусного сходства

    # Нормализуем эмбединги для корректной работы с cosine similarity
    faiss.normalize_L2(embeddings)
    self.index.add(embeddings.astype('float32'))

def search(self, query: str, k: int = 3) -> List[str]:
    """Ищем наиболее релевантные документы"""
    # Векторизуем запрос
    query_embedding = self.encoder.encode([query])
    faiss.normalize_L2(query_embedding)

    # Выполняем поиск в FAISS
    scores, indices = self.index.search(query_embedding.astype('float32'), k)

    # Возвращаем найденные документы
    return [self.documents[idx] for idx in indices[0]]

def generate_answer(self, query: str, context: List[str]) -> str:
    """Генерируем ответ на основе найденного контекста"""
    # Формируем промпт с контекстом
    context_text = "\n".join([f"Документ {i+1}: {doc}" for i, doc in
enumerate(context)])

    prompt = f"""Контекст:
{context_text}

Вопрос: {query}

Ответ на вопрос, используя только информацию из предоставленного контекста.
Если ответа нет в контексте, так и напиши."""

    # Вызываем OpenAI API
    response = openai.ChatCompletion.create(
        model="gpt-3.5-turbo",
        messages=[{"role": "user", "content": prompt}],
        max_tokens=200,
        temperature=0.1
    )

    return response.choices[0].message.content.strip()

def ask(self, query: str) -> dict:
    """Основная функция: поиск + генерация ответа"""
    # Находим релевантные документы
    relevant_docs = self.search(query)

```

```
# Генерируем ответ
answer = self.generate_answer(query, relevant_docs)

return {
    "query": query,
    "answer": answer,
    "sources": relevant_docs
}
```

Демонстрация работы

Теперь создадим базу знаний и протестируем систему:

```
# Создаём экземпляр RAG-системы
rag = SimpleRAG(api_key="your-openai-api-key")

# Добавляем документы в базу знаний
documents = [
    "Python – высокоуровневый язык программирования общего назначения. Создан в 1991 году [Гвидо ван Россумом.",
    "RAG (Retrieval-Augmented Generation) – архитектурный подход, который объединяет поиск информации с генерацией текста.",
    "Векторные базы данных хранят данные в виде высокомерных векторов и позволяют выполнять семантический поиск.",
    "FAISS – библиотека для эффективного поиска схожих векторов.",
    "Эмбединги – это векторные представления текста, которые кодируют семантическое содержание в числовой форме."
]

rag.add_documents(documents)

# Задаём вопросы системе
questions = [
    "Кто создал Python?",
    "Что такое RAG?",
    "Как работают векторные базы данных?"
]

for question in questions:
    result = rag.ask(question)
    print(f"Вопрос: {result['query']}")
    print(f"Ответ: {result['answer']}")
    print(f"Источники: {result['sources']}")
    print("-" * 50)
```

Разбор ключевых моментов

Векторизация документов происходит в методе `add_documents()`. Мы используем предобученную модель **all-MiniLM-L6-v2**, которая преобразует тексты в 384-мерные векторы, эффективно кодирующие семантическое содержание.

Индекс FAISS IndexFlatIP выполняет поиск на основе внутреннего произведения векторов. После нормализации это эквивалентно косинусному сходству – стандартной метрике для сравнения семантической близости текстов.

Формирование промпта в методе `generate_answer()` следует классическому паттерну RAG: контекст + вопрос + инструкция. Языковая модель получает найденную информацию и генерирует ответ, ограничиваясь предоставленным контекстом.

Ограничения и возможности развития

Этот минимальный пример демонстрирует базовые принципы, но в реальных системах потребуются дополнительные компоненты:

- сегментирование длинных документов на фрагменты оптимального размера;
- предварительная обработка текста (очистка, нормализация);
- гибридный поиск, комбинирующий семантический и лексический поиск;
- ранкинг найденных фрагментов для повышения точности;
- обработка ошибок и валидация входных данных.

Тем не менее даже эта простая реализация способна давать осмысленные ответы на вопросы по загруженным документам, демонстрируя мощь RAG-подхода. Система понимает семантику запросов, находит релевантную информацию и формирует связные ответы – всё то, что делает RAG революционной технологией в области обработки естественного языка.

Главная ценность этого примера – в наглядности принципов работы RAG. Понимание того, как векторы кодируют смысл, как работает семантический поиск и как контекст влияет на генерацию, – всё это даёт прочную основу для создания более сложных и эффективных систем в следующих главах книги.