Сравнение алгоритмов семантического разбиения текстов научных статей для извлечения знаний

С. А. Лавров

Санкт-Петербургский государственный университет аэрокосмического приборостроения

lav.svi@yandex.ru

М. Д. Поляк

Санкт-Петербургский государственный университет аэрокосмического приборостроения

markpolyak@gmail.com

Аннотация. **В данной работе проведено сравнительное исследование пяти алгоритмов семантического сегментирования (чанкинга) научных статей — PGC, PCC, DFC, SSTC и LBDC — как критического этапа в архитектуре систем Retrieval-Augmented Generation (RAG). С использованием экспериментальной платформы и наборов данных PeerQA и Qasper проведен многофакторный анализ влияния стратегий разбиения на внутренние метрики связности, вычислительную сложность и итоговое качество генерации ответов. Установлено, что иерархический метод PCC обеспечивает наилучшую полноту извлечения знаний (Context Recall до 0.78), в то время как семантические методы SSTC и LBDC позволяют существенно повысить верность ответов (Faithfulness) и снизить операционные затраты за счет оптимизации объема входных токенов. Выявлен критический порог латентности для методов на базе LLM, ограничивающий их применение в системах реального времени. Сформулированы практические рекомендации по выбору стратегии сегментирования в зависимости от приоритетов системы: полноты, достоверности или экономической эффективности.**

Ключевые слова: семантическое сегментирование; чанкинг; информационный поиск; научные статьи; обработка естественного языка; метрики оценки; генерация с дополненной выборкой

I. Введение

Современные системы автоматизированного анализа научной литературы, построенные на парадигме генерации с дополненной выборкой (Retrieval-Augmented Generation, RAG), демонстрируют высокий потенциал в решении задач информационной поддержки исследовательской деятельности. Фундаментальным этапом в конвейере таких систем является индексирование, в ходе которого исходные полнотекстовые документы преобразуются в семантически целостные фрагменты — чанки (chunks).

Качество сегментирования напрямую определяет эффективность всей последующей работы системы [1]: избыточно крупные фрагменты приводят к зашумлению контекста и росту операционных затрат, в то время как чрезмерно мелкие чанки могут привести к потере ключевых логических связей и снижению полноты ответа. Особую сложность представляет обработка научных статей, характеризующихся высокой плотностью терминологии, сложной логической структурой и наличием специфических связей между разделами.

Несмотря на наличие множества подходов к разбиению текста — от простейших фиксированных окон до сложных моделей на базе больших языковых моделей (LLM) — вопрос выбора оптимальной стратегии для работы с научным дискурсом остается открытым. В существующих работах [2, 3] часто игнорируется баланс между вычислительной сложностью алгоритма сегментации и итоговой стоимостью эксплуатации RAG-системы.

В данной статье представлено систематическое сравнение пяти разноплановых алгоритмов: структурных (PGC, PCC), адаптивных (DFC) и семантически-ориентированных (SSTC, LBDC). Целью работы является выявление закономерностей влияния методов сегментирования на метрики полноты поиска (Context Recall), верности генерации (Faithfulness) и на экономическую эффективность системы. В ходе исследования используется специализированная экспериментальная платформа и верифицированные наборы данных научных вопросов PeerQA [4] и Qasper [5], что позволяет оценить применимость каждого подхода в условиях реальных исследовательских задач.

II. Формализация задачи

Пусть дан документ D, представленный как последовательность предложений $s_1, s_2, ..., s_n$. Задача семантического разбиения состоит в нахождении множества индексов границ $B \subset \{1, ..., n-1\}$, которое делит D на упорядоченный набор чанков $C = \{c_1, c_2, ..., c_k\}$. Оптимальное разбиение B* минимизирует внутрикластерную семантическую дивергенцию и максимизирует межкластерную:

$$B^* = \alpha \cdot \text{intra}(B) - \beta \cdot \text{inter}(B),$$

где $\text{intra}(B) = \Sigma\, \text{avg\_cos\_sim}(s_i, s_j)$ для $s_i, s_j \in c_k$, а $\text{inter}(B) = \text{avg\_cos\_sim}(\text{last}(c_k), \text{first}(c_{k+1}))$ по всем смежным парам чанков; α, β — весовые коэффициенты компромисса. Дополнительными ограничениями для научного текста служат: максимальный размер чанка в токенах T_max, минимальный T_min, а также наличие структурных элементов (формулы, таблицы, заголовки разделов), нарушение которых недопустимо при проведении границы.

III. Классификация

Существующие подходы к сегментации текста можно разделить на 5 широких категорий [2]: детерминированные, рекурсивные, семантические,

адаптивные, а также гибридные. Каждая категория представляет собой концепцию разделения текста на фрагменты.

A. *Детерминированные*

Данный класс методов основан на заранее заданных правилах, таких как фиксированный размер фрагмента, разбиение по предложениям или абзацам. Их основное преимущество заключается в простоте реализации и предсказуемости поведения. Однако такие методы не учитывают семантические границы текста, что часто приводит к разрыву логически связанных фрагментов.

B. *Рекурсивные*

В основе этих методов лежит идея разбиения текста на вложенные структуры, отражающие его иерархическую организацию. Алгоритм последовательно делит текст на крупные сегменты, а затем рекурсивно уточняет разбиение на более мелкие части, формируя дерево сегментов. Такие подходы хорошо отражают структурную композицию текста (разделы, подразделы), но требуют выбора критериев разбиения и могут быть чувствительны к шуму.

C. *Семантические*

В данных подходах сегментация выполняется на основе семантического сходства между частями текста. Обычно используются векторные представления предложений, а границы определяются в точках резкого изменения темы. Эти методы позволяют выделять когерентные фрагменты, но чувствительны к выбору модели эмбеддингов.

D. *Адаптивные*

Адаптивные методы изменяют стратегию сегментации в зависимости от характеристик текста, таких как плотность информации или вариативность тем. Они стремятся находить баланс между длиной сегмента и его информативностью, однако требуют дополнительных критериев и параметров настройки.

E. *Гибридные*

Гибридные методы объединяют несколько подходов, например, используют эвристическое разбиение с последующей семантической корректировкой. Это позволяет компенсировать недостатки отдельных методов и добиться более устойчивых результатов, особенно в условиях неоднородных текстов.

Таким образом, различные категории алгоритмов сегментирования представляют собой компромисс между вычислительной эффективностью и качеством выделения семантически целостных фрагментов.

IV. Специфика предметной области

Как показал предварительный анализ, применение идей сегментации текста в чистом виде для обработки научных статей не является корректным решением: специфика работы с научным текстом требует не просто механического разделения, а глубокой адаптации к его сверхплотной информационной архитектуре.

Научная статья — это предельно насыщенный и огромный по размерам блок данных, где концентрация смысла на один абзац текста в разы превышает таковую в публицистике или технической документации.

Поэтому мы внедряем жесткое ограничение по токенам на чанк и выверенный overlap для каждого алгоритма разбиения. Это позволяет искусственно поддерживать контекстную связность на границах фрагментов, не допуская потери ключевых смысловых цепочек, а также дает возможность корректного сравнения чанкеров.

Такой минимальный подход позволяет адаптировать теоретические идеи сегментации для работы с аномально сложным контентом в условиях ограниченных вычислительных ресурсов. В конечном итоге это максимально приближает нас к реалистичным условиям эксплуатации RAG-систем, где точность извлечения факта из плотного научного массива является приоритетом.

V. Исследуемые алгоритмы

В рамках адаптации ключевых идей сегментирования текста в связке с ограничением по токенам и overlap были реализованы гибридные чанкеры с максимальным сохранением ключевой идеи метода:

A. *Paragraph Group Chunking (PGC) – A + B + D*

Алгоритм пытается разбить текст по самым крупным разделителям, и если полученная часть вместе с overlap (40 токенов) от предыдущего куска превышает лимит (450 токенов), он рекурсивно спускается к более мелким разделителям — вплоть до отдельных слов или жесткой нарезки по токенам.

B. *Parent–Child Chunking (PCC) – B + A*

Этот чанкер реализует стратегию динамического контекстного окна для систем RAG: он нарезает текст на мелкие «детские» фрагменты для точного векторного поиска, но для каждого из них формирует расширенный «родительский» контекст. Процесс начинается с дробления документа на небольшие части заданного размера (150 токенов) с overlap (40 токенов), после чего алгоритм находит положение каждого такого куска в исходном тексте и симметрично расширяет его границы до более крупного окна (400 токенов). В итоге в поисковый индекс попадает краткий фрагмент, а в его метаданные записывается окружающий текст, что позволяет нейросети видеть полную смысловую картину.

C. *Semantic Similarity Threshold Chunking (SSTC) – C + D*

Чанкер разбивает текст, основываясь на изменении смысла между предложениями. Сначала он делит документ на отдельные предложения и вычисляет для них векторные представления (эмбеддинги), затем сравнивает группы предложений (4 предложения) слева и справа от каждой границы. Если косинусное сходство между соседними окнами падает ниже адаптивного порога (процентиль 60), это сигнализирует о смене темы, и чанкер создает разрыв. Дополнительно алгоритм соблюдает жесткие ограничения по минимальному (2 предложения) и максимальному (20 предложений) количеству предложений, чтобы чанки не получались слишком мелкими или огромными, и поддерживает overlap (2 предложения) для сохранения связности между блоками.

D. *Dynamic Token Size Chunking (DFC) – D + A*

Этот метод реализует стратегию адаптивной токенизации, которая динамически подстраивает размер каждого фрагмента под сложность текста, предотвращая

переполнение контекстного окна. Алгоритм сначала разбивает текст на предложения с помощью регулярных выражений, затем токенизирует их и вычисляет «лексическую плотность» (соотношение уникальных токенов к общему количеству). На основе этой плотности он пересчитывает целевой размер чанка (200 токенов): если текст перенасыщен уникальными словами, чанкер уменьшает объем фрагмента (минимальный размер 100 токенов), чтобы не перегружать модель, а если текст простой — увеличивает его (максимальный размер 400 токенов). Весь процесс сопровождается строгим контролем токенов и формированием overlap (40 токенов).

E. LLM Boundary Detection (LBDC) – C + D + E

Этот чанкер представляет собой двухэтапную систему семантического анализа. На первом этапе текст разбивается на предложения, и алгоритм вычисляет косинусное сходство между их эмбеддингами; если сходство падает ниже порога (0.75) или встречается заголовок, граница помечается как подозрительная. На втором, критическом этапе, все сомнительные границы отправляются в LLM: модель анализирует контекст трех предыдущих предложений и решает, действительно ли здесь начинается новая логическая тема. В завершение чанкер собирает текст в итоговые блоки, добавляя фиксированный overlap (2 предложения).

Параметры каждого алгоритма подбирались эмпирически с учётом 3 факторов: (1) ограничений целевых моделей (эмбеддер и LLM имеют ограничение на контекстное окно в 4096 токенов); (2) природы научного текста (длинные предложения, формулы, ссылки, заголовки); (3) компромисса между гранулярностью и связностью.

VI. RAG-система и модели

Каждый алгоритм был инсталлирован в RAG-пайплайн [6]: Docling для парсинга текста научных статей в markdown формате; векторная база данных FAISS, выступающая в роли ретривера; эмбеддер модель Qwen3-Embedding-0.6B для создания вектора смыслового представления фрагмента текста; LLM Qwen2.5-7B-Instruct-GPTQ-Int4 с 4 битным квантованием на vllm движке, использующаяся в качестве генератора ответов, а также ее токенизатор для преобразования входной последовательности символов в токены.

Для RAG-метрик были применены LLM Meta-Llama-3.1-8B-Instruct-GPTQ-INT4 с 4 битным квантованием на движке vllm с bge-m3 эмбеддером.

RAG-система запускалась на двух gpu с ограничением vRAM равным 15 гб на одну видеокарту.

VII. Данные и метрики

Корпус данных для оценки качества работы чанкеров представляет собой набор 200 статей из arXiv, отобранных по категориям: physics.soc-ph (40), cs.CL (40), math.CO (40), stat.ML (34), q-bio.QM (6). Общее количество токенов – около 2 млн.

Датасеты для RAG-оценки представляют собой подвыборку QA-датасетов Qasper (925) [5] и PeerQA (87) [4] с эталонными ответами. Отбор данных происходил по наличию готового ответа на вопрос и фрагментов текста исходной статьи, в которых содержится ответ.

Используемые в статье метрики можно разделить на три группы: внутренние, внешние и RAG-метрики.

Внутренние – оценивают качество разбиения без привязки к задаче:

- средняя косинусная близость между соседними предложениями внутри чанка (avg_sent_sim);
- средняя длина чанка в токенах (avg_chunk_len_tokens);
- связность между чанками (косинусная близость граничных предложений) – connectivity;
- среднее число чанков на документ (chunks_per_doc).

Внешние – производительность:

- среднее, 95-й перцентиль, минимум и максимум времени обработки одного документа.

RAG-метрики – оценка влияния разбиения на QA-систему:

- context relevance (релевантность извлечённых чанков вопросу);
- faithfulness (соответствие ответа контексту);
- context recall (полнота извлечения);
- число входных/выходных токенов, нормализованная стоимость инференса (рассчитана по тарифу API GPT-4o-mini как эталонная единица для сопоставления токенной нагрузки между методами);
- общая латентность (retrieval + generation).

Расчёт RAG-метрик производился с помощью фреймворка Ragas.

VIII. Результаты

Табл. 1 описывает технические характеристики самих фрагментов текста (чанков) и скорость работы алгоритмов.

В табл. 2 приведены результаты работы всей вопросно-ответной системы, где оценивается, как выбранный способ разбиения текста влияет на итоговый ответ нейросети. Метрики faithfulness, context recall и context relevance представлены как среднее значение метрики.

ТАБЛИЦА I. ВНУТРЕННИЕ МЕТРИКИ И ПРОИЗВОДИТЕЛЬНОСТЬ

Метод	Avg Sent Sim	Avg Chunk Len Tokens	Connectivity	Chunks Per Doc	Latency Mean	Latency P95	Latency Min	Latency Max	Norm Latency 1K Tokens	Norm Connectivity Per 100 Tokens
PGC	0.9611	367.29	0.7056	26.8	0.0863s	0.1865s	0.0107s	0.3006s	0.0088s	0.1921
PCC	0.9692	382.1	0.7954	274.65	0.0089s	0.021s	0.0013s	0.0389s	0.0001s	0.2082
DFC	0.9185	193.55	0.7165	52.91	0.6991s	0.7332s	0.6592s	0.7537s	0.0683s	0.3702
SSTC	0.7953	107.03	0.79	189.97	2.3s	3.66s	0.9603s	5.45s	0.1131s	0.7381
LBDC	0.7849	97.04	0.7945	236.06	28.4s	57.17s	4.03s	78.92s	1.24s	0.8188

ТАБЛИЦА II. RAGAS-МЕТРИКИ

Метод	faithfulness	context_recall	context_relevance	Norm. Cost (USD)	Input Tokens	Output Tokens	Total Tokens	Latency P95	Total Sec
PeerQA									
PGC	0.665152	0.709903	0.933908	0.020474	102,773	8,430	111,203	1.26s	109.61s
PCC	0.679327	0.706841	0.867816	0.018688	92,317	8,068	100,385	1.13s	98.01s
DFC	0.637593	0.708206	0.899425	0.012754	52,892	8,033	60,925	0.78s	67.94s
SSTC	0.642489	0.690772	0.913793	0.008998	32,720	6,816	39,536	0.55s	47.59s
LBDC	0.625801	0.708627	0.916667	0.008569	30,627	6,625	37,252	0.52s	44.98s
Qasper									
PGC	0.724906	0.731347	0.871892	0.205447	1,081,521	72,032	1,153,553	1.09s	1000.75s
PCC	0.735229	0.784773	0.841081	0.190911	1,003,251	67,372	1,070,623	1.02s	938.87s
DFC	0.728285	0.729613	0.872162	0.130166	602,619	66,289	668,908	0.66s	612.25s
SSTC	0.748271	0.581822	0.874054	0.084082	342,246	54,575	396,821	0.43s	395.65s
LBDC	0.732630	0.578389	0.874324	0.079870	316,094	54,093	370,187	0.41s	381.59s

IX. ОБСУЖДЕНИЕ

Результаты проведенных экспериментов позволяют провести комплексный анализ влияния стратегий сегментирования на производительность и качество систем RAG при работе с научными текстами. Полученные данные (табл. 1 и табл. 2) демонстрируют наличие фундаментального компромисса между семантической точностью разбиения и вычислительными затратами на этапе индексирования.

A. Анализ характеристик сегментирования и вычислительной сложности

Проведенный количественный анализ (табл. 1) выявляет существенную разницу в поведении методов сегментирования. У каждой стратегии свой баланс между глубиной семантического анализа и требуемыми вычислительными ресурсами.

Метод PGC выступает в качестве базового алгоритма. Он характеризуется высокими показателями средней схожести предложений внутри фрагмента (Avg Sent Sim = 0.96). Это объясняется тем, что алгоритм группирует параграфы до достижения лимита токенов без глубокого анализа тематических переходов. К преимуществам данного подхода можно отнести высокую скорость обработки (Latency Mean = 0.009 с) и сохранение широкого контекста, однако это приводит к избыточности данных. Средняя длина чанка составляет 370 токенов, это почти самое высокое значение среди исследуемых алгоритмов, следовательно можно сделать вывод о том, что чанкер включает в себя информацию, не относящуюся к конкретному запросу пользователя, а это приводит к увеличению затрат на инференс LLM (табл. 2, Cost).

Алгоритм PCC реализует иерархическую стратегию, где поиск введется по малым фрагментам, а в модель генерации передаются более крупные блоки. По показателям латентности PCC сопоставим с PGC, являясь одним из самых быстрых методов из обозреваемых. При этом он сохраняет высокую связность (Connectivity = 0.8), что критично для научных текстов с перекрестными ссылками. Данный метод минимизирует риск потери контекста, однако, как и PGC, страдает от высокой семантической избыточности внутри «родительских» блоков.

DFC представляет собой попытку сбалансировать структурный и семантический подходы. Алгоритм варьирует размер фрагмента в зависимости от плотности ключевых терминов и структуры документа. Такой подход является более время затратным (Latency Mean = 0.6991 с), чем PGC или PCC, но на порядок ниже, чем нейросетевой подход. За большее время выполнения данный алгоритм генерирует умеренное количество чанков на документ (Chunks Per Doc = 52.91), обеспечивая оптимальное покрытие материала без чрезмерного дробления текста, что делает его «золотой серединой» для систем с ограниченными вычислительными ресурсами.

Метод SSTC базируется на анализе косинусного расстояния между векторами эмбеддингов соседних предложений. Граница чанка проводится там, где семантическая близость опускается ниже адаптивного порога. Этот алгоритм демонстрирует заметно более низкую внутреннюю схожесть (Avg Sent Sim = 0.7953). Это указывает на то, что фрагменты получаются тематически изолированными и специфичными, однако такое преимущество влечет за собой ощутимое возрастание задержки из-за необходимости генерации эмбеддингов для каждого предложения, пусть и итоговая стоимость работы RAG-системы снижается за счет передачи в LLM только компактных и релевантных данных.

LBDC является наиболее технологически сложным методом, где решение о разрыве контекста принимает языковая модель на основе анализа логики повествования. Метод демонстрирует экстремально высокую латентность – 28.4 секунды на один документ.

Такой показатель делает невозможным использование LBDC в высоконагруженных системах без предварительной офлайн-индексации. С другой стороны, показатель средней длинны чанка составляет порядка 100 токенов при сохранении высокого уровня связности (Connectivity = 0.7945). Это позволяет системе извлекать предельно точные ответы на узкоспециализированные вопросы, что подтверждается высокими показателями по метрике Faithfulness в табл. 2.

Сравнительный анализ подтверждает, что увеличение вычислительной сложности (от PGC к LBDC) напрямую коррелирует с ростом семантической точности сегментации. Если для оперативных задач поиска подходят PGC и PCC, то для глубокого анализа научной литературы, где критична чистота контекста и минимизация галлюцинаций, предпочтительны SSTC и DFC, обеспечивающие наилучшее соотношение «скорость — качество».

B. Влияние стратегий на качество поиска и генерации

Анализ результатов работы RAG-системы (табл. 2) позволяет оценить прикладную эффективность каждого алгоритма сегментирования. Мы наблюдаем существенную вариативность метрик в зависимости от архитектуры чанкера и сложности исходного набора данных (PeerQA vs Qasper).

Алгоритм PGC демонстрирует стабильно высокие показатели полноты извлечения (Context Recall) на обоих датасетах: 0.7099 для PeerQA и 0.7313 для Qasper. Высокая релевантность контекста (Context Relevance = 0.9339) на PeerQA объясняется тем, что крупные фрагменты (в среднем 367 токенов) практически гарантированно содержат искомый факт вместе с его окружением. Обратной стороной является самая высокая нормализованная стоимость инференса (0.205 USD на Qasper) и избыточная нагрузка на контекстное окно модели (более 1 млн входных токенов на корпус Qasper).

Метод PCC показал себя как наиболее эффективный инструмент для поиска информации в сложных структурах. На датасете Qasper он достиг наивысшего показателя Context Recall (0.7847) среди всех исследуемых методов. Преимущество достигается за счет разделения функций: мелкие «детские» фрагменты обеспечивают высокую точность векторного совпадения, а расширенный «родительский» контекст позволяет модели генерации точнее формулировать ответ. Метод также хорошо себя показывает по метрике Faithfulness (0.7352) на Qasper, что подтверждает гипотезу о важности широкого контекста для минимизации галлюцинаций LLM при обработке научных данных.

DFC выступает в роли сбалансированного решения. Его показатели полноты (0.7082) и верности (0.6375) на PeerQA сопоставимы с лидерами, но при значительно меньших затратах. Адаптация размера чанка под лексическую плотность текста позволила сократить количество входных токенов почти вдвое по сравнению с PGC (52,892 против 102,773 на PeerQA). Снижение объема данных напрямую отразилось на латентности генерации (Latency P95: 0.78s), что делает DFC предпочтительным для систем с высокими требованиями к скорости отклика.

Семантический метод SSTC демонстрирует уникальный профиль эффективности. На датасете Qasper он показал абсолютный максимум по метрике Faithfulness (0.7482). Точное определение семантических границ между темами позволяет изолировать нужную информацию от побочных данных. Это исключает ситуацию, когда модель пытается объединить в ответе факты из разных смысловых блоков статьи. SSTC является одним из самых дешевых методов в эксплуатации — общая стоимость обработки PeerQA составила всего 0.0089 USD. Однако за это приходится платить снижением полноты поиска (Context Recall падает до 0.58 на Qasper) из-за предельно малого размера чанков.

Метод LBDC показывает результаты, близкие к SSTC, но с еще большим уклоном в экономию токенов. Он обеспечивает минимальное использование ресурсов (Input Tokens: 30,627 на PeerQA) и самую низкую латентность этапа генерации (0.52s). Высокая релевантность контекста (0.9166) подтверждает, что использование LLM на этапе сегментации позволяет выделять максимально «чистые» смысловые единицы. Как и SSTC, метод страдает от низкого охвата на сложных данных (Recall: 0.5783 на Qasper). Это указывает на то, что для глубоких научных ответов «чистоты» маленького чанка (97 токенов) часто бывает недостаточно без привлечения дополнительного контекста.

Таким образом, выбор алгоритма сегментирования напрямую определяет баланс между точностью ответа и стоимостью системы. В то время как PCC является неоспоримым лидером по качеству извлечения знаний из плотных научных текстов, семантические методы (SSTC, LBDC) предлагают беспрецедентную экономическую эффективность для задач, где допустимо пожертвовать полнотой ради скорости и верности единичного факта.

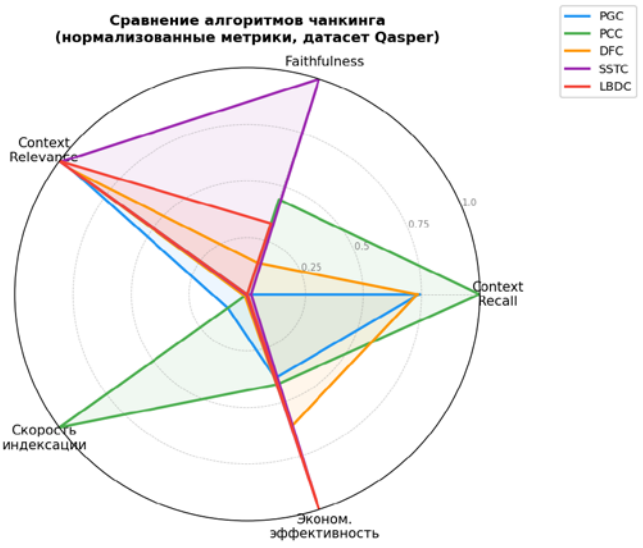

Рис. 1. Сравнение алгоритмов чанкинга по нормализованным метрикам (датасет Qasper)

C. Рекомендации по выбору стратегии

На основе сопоставления внутренних метрик качества разбиения и итоговой производительности RAG-системы, можно сформулировать дифференцированные рекомендации по выбору алгоритма чанкинга в зависимости от технических ограничений и бизнес-целей системы.

Для исследовательских инструментов и экспертных систем, где критически важно не упустить ни одного факта из первоисточника (особенно на сложных датасетах типа Qasper), оптимальным выбором является алгоритм Parent–Child Chunking (PCC).

В задачах, требующих строгой верификации ответов по научным источникам, наиболее эффективными показали себя семантические методы, в частности Semantic Similarity Threshold Chunking (SSTC). Использование LLM-Based Decision Chunking (LBDC) также оправдано в случаях, когда требуется исключительная «чистота» границ, а общее количество документов невелико.

Для крупномасштабных систем с миллионами запросов критическим фактором становится стоимость инференса и количество токенов. Алгоритмы SSTC и LBDC позволяют сократить нормализованную стоимость более чем в 2.5 раза (до 0.008–0.079 USD за цикл обработки) по сравнению с базовым методом PGC за счет передачи в LLM только компактных и высокорелевантных данных.

Если система требует мгновенной индексации новых поступлений (например, мониторинг препринтов), использование методов на базе LLM (LBDC) или тяжелых эмбеддеров (SSTC) становится нецелесообразным из-за задержек (до 28.4 с на документ). В таких условиях рекомендуется использовать PCC или PGC, чья латентность не превышает 0.009–0.08 с, что позволяет обрабатывать тысячи страниц в минуту без значительных вычислительных мощностей.

С точки зрения баланса «цена-качество», наиболее перспективным выглядит Dynamic Token Size Chunking (DFC), который обеспечивает двукратную экономию токенов при сохранении высокой скорости работы и приемлемой точности.

X. Ограничения исследования

Настоящее исследование проводилось в условиях ограниченных вычислительных ресурсов (бесплатные GPU-квоты платформ Kaggle и Google Colab, не более 15 ГБ vRAM на карту), что обусловило выбор моделей меньшего размера — Qwen2.5-7B-Instruct и Qwen3-Embedding-0.6B. Полученные результаты могут не переноситься напрямую на более мощные модели; проверка чувствительности к выбору эмбеддера и генератора обозначена авторами как приоритетное направление дальнейшей работы.

Корпус данных (200 статей arXiv) характеризуется неравномерным распределением по предметным областям: категория q-bio.QM представлена лишь 6 документами, что не позволяет делать обобщённые выводы о применимости методов в биомедицинских и смежных областях.

Стоимость инференса в табл. 2 приведена в виде нормализованной оценки, рассчитанной по публичному тарифу API GPT-4o-mini и предназначенной исключительно для сопоставления относительной токенной нагрузки между методами; реальные затраты при локальном развёртывании определяются используемым оборудованием.

XI. Заключение

В рамках данного исследования было проведено комплексное сравнение пяти алгоритмов семантического разбиения текстов научных статей (PGC, PCC, DFC, SSTC и LBDC) в контексте их применения в архитектуре Retrieval-Augmented Generation (RAG). На основе проведенного тестирования на наборах данных PeerQA и Qasper были сформулированы следующие выводы:

- Выявлен фундаментальный компромисс между вычислительной сложностью алгоритма на этапе предобработки (индексации) и операционными затратами на инференс больших языковых моделей.
- Структурно-иерархические методы (в частности, PCC) доказали свою высокую эффективность в сценариях, требующих максимальной полноты извлечения знаний из плотного научного дискурса. Алгоритм PCC обеспечил наилучший показатель Context Recall (0.7847) на сложном датасете Qasper при минимальной латентности сегментации.
- Методы семантического анализа границ (SSTC и LBDC) позволяют существенно изолировать релевантный контекст, что прямо отражается на росте верности ответов (Faithfulness) и снижении стоимости генерации за счет двукратного сокращения объема входных токенов. Однако высокая задержка метода LBDC (до 28.4 с на документ) ограничивает его применение офлайн-режимом.
- Метод DFC приобрел статус адаптивного компромиссного решения, способного динамически балансировать нагрузку в зависимости от локальной лексической плотности текста.

В качестве перспективных направлений дальнейшей работы авторы видят:

- Исследование чувствительности данных методов к смене моделей эмбеддингов (по аналогии с подходами, предложенными в последних фундаментальных бенчмарках 2026 года [7]).
- Адаптацию алгоритмов под мультимодальные научные документы, содержащие не только текст, но и формулы, графики и таблицы [8].
- Оптимизацию вызовов LLM в алгоритме LBDC с целью снижения временных затрат без потери качества семантического разбиения.

Список литературы

[1] Jintao Zhang, Guoliang Li, Jinyang Su. SAGE: A Framework of Precise Retrieval for RAG. 2025 IEEE 41st International Conference on Data Engineering (ICDE), Hong Kong, Hong Kong, 2025, pp. 1388-1401, doi: 10.1109/ICDE65448.2025.00108.

[2] Muhammad Arslan Shaukat, Muntasir Adnan, Carlos C. N. Kuhn. A Systematic Investigation of Document Chunking Strategies and Embedding Sensitivity. – 2026 [Электронный ресурс]. URL: https://arxiv.org/abs/2603.06976v1 (дата обращения: 12.03.2026).

[3] Omri Koshorek, Adir Cohen, Noam Mor, Michael Rotman, Jonathan Berant. Text Segmentation as a Supervised Learning Task – 2018 [Электронный ресурс]. URL: https://arxiv.org/abs/1803.09337 (дата обращения: 26.02.2026)

[4] Luis Chiruzzo, Alan Ritter, Lu Wang. PeerQA: A Scientific Question Answering Dataset from Peer Reviews. – 2025 [Электронный ресурс]. URL: https://aclanthology.org/2025.naacl-long.22.pdf (дата обращения 26.02.2026)

[5] Pradeep Dasigi, Kyle Lo, Iz Beltagy, Arman Cohan, Noah A. Smith, Matt Gardner. A Dataset of Information-Seeking Questions and Answers Anchored in Research Papers. – 2021 [электронный ресурс]. URL: https://arxiv.org/abs/2105.03011 (дата обращения: 26.02.2026).

[6] Agada Joseph Oche, Ademola Glory Folashade, Tirthankar Ghosal, Arpan Biswas. A Systematic Review of Key Retrieval-Augmented Generation (RAG) Systems: Progress, Gaps, and Future Directions. – 2025 [Электронный ресурс]. URL: https://doi.org/10.48550/arXiv.2507.18910 (дата обращения 13.02.2026).

[7] Wensheng Lu, Keyu Chen, Ruizhi Qiao, Xing Sun. HiChunk: Evaluating and Enhancing Retrieval-Augmented Generation with Hierarchical Chunking. – 2025 [Электронный ресурс]. URL: https://arxiv.org/abs/2509.11552v3 (дата обращения: 12.03.2026).

[8] Jihao Zhao, Daixuan Li, Pengfei Li, Shuaishuai Zu, Biao Qin, Hongyan Liu. QChunker: Learning Question-Aware Text Chunking for Domain RAG via Multi-Agent Debate. – 2026 [Электронный ресурс]. URL: https://arxiv.org/abs/2603.11650v1 (дата обращения: 18.03.2026)