Универсальный механизм первичного поиска повторов в тексте для пакета Duplicate Finder

Глазырин Антон Георгиевич

Научный руководитель: доц. каф. СП, к.ф-м.н. Луцив Д. В.

Рецензент: ген. дир. ООО «Ембокс» Бондарев А. В.

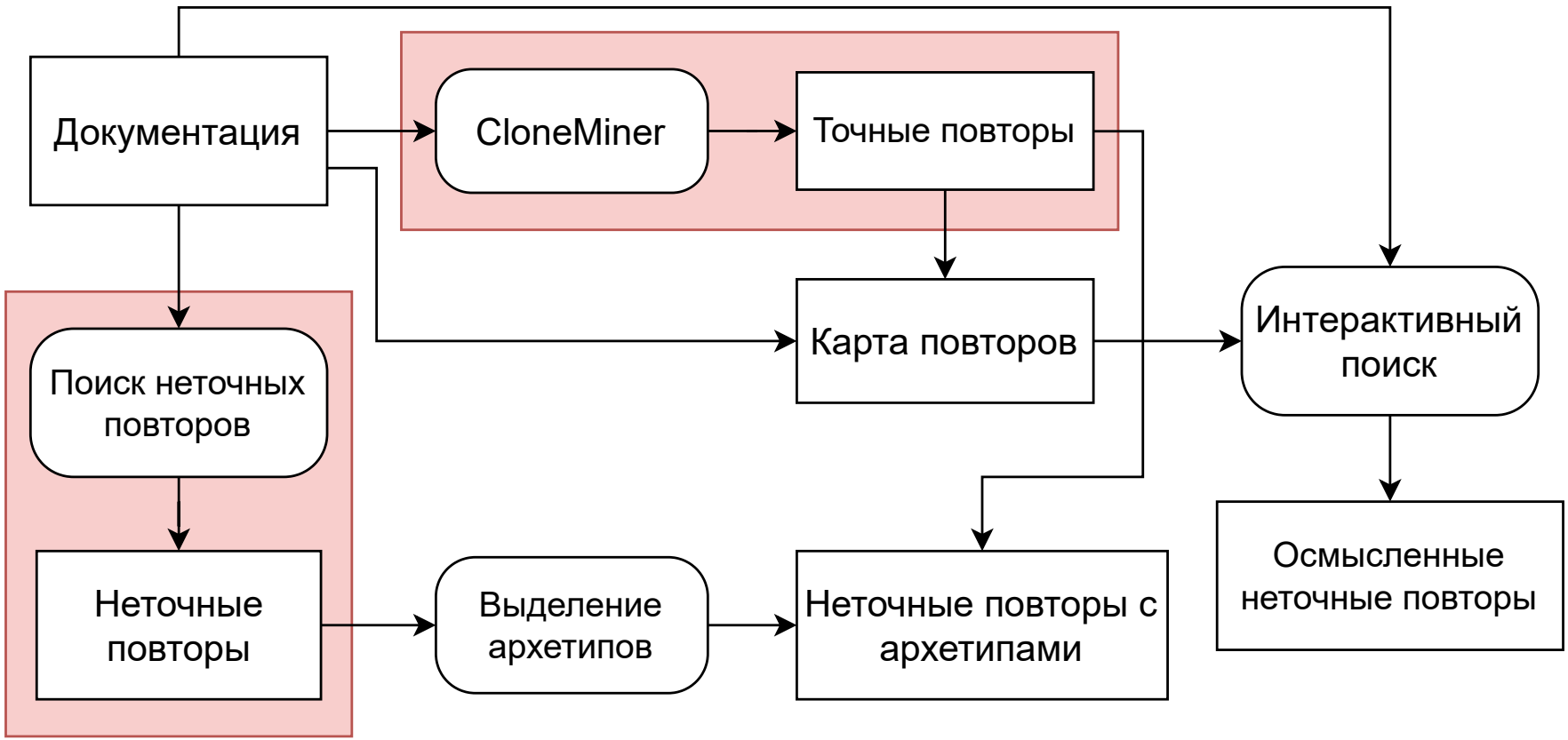
СПбГУ

6 июня 2023 г.

Мотивация

- E. Juergens, J. Porubän: $\approx$10% документации — дублированные фрагменты
- Негативное влияние повторов:
  - Раздувание объема
  - Усложнение модификации
- Управление повторами — улучшение документации

Мотивация: Duplicate Finder



Постановка задачи

Цель — разработка унифицированной подсистемы поиска точных и неточных повторов для Duplicate Finder Toolkit.

- Анализ предметной области
- Определение проблем поиска повторов в DuplicateFinder и требований к новому механизму
- Проектирование конвейера механизма поиска повторов
- Разработка алгоритмов точного и неточного поиска
- Реализация инструмента и его интеграция в DuplicateFinder
- Проведение тестирования разработанного инструмента

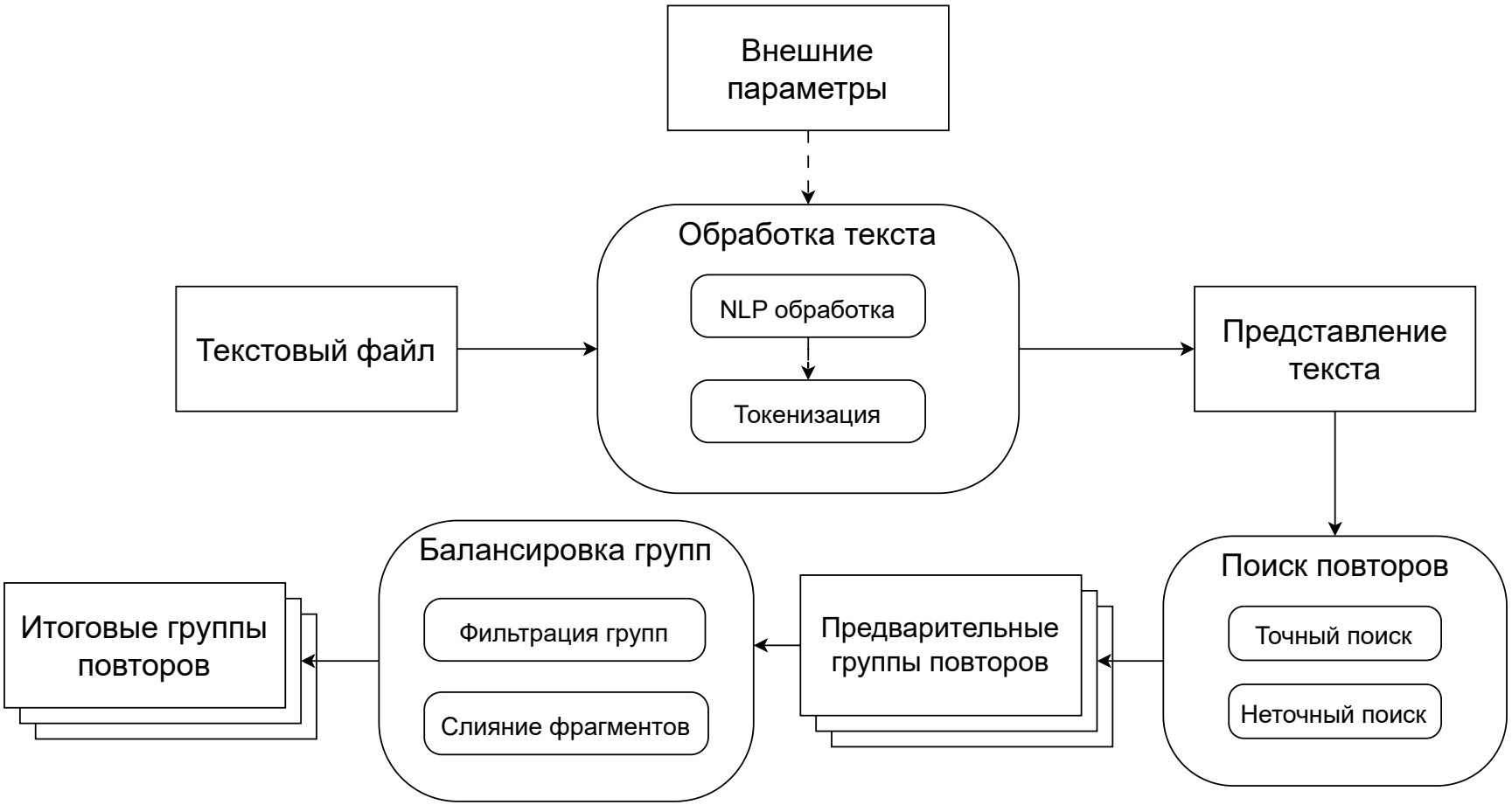
Существующие решения

- Поиск клонов в ПО:
  - CCFinder
  - CloneMiner
  - Klocwork inSight
  - cpdetector
- Сравнение текстовых документов:
  - Align
  - TxtAlign
- Поиск по образцу:
  - Duplicate Defect Detection
  - Apache Lucene
  - FactorLCS

Определение требований

1. Реализация на языке Python
2. Открытая разработка
3. Поиск точных и неточных повторов
4. Универсальность процесса поиска
5. Наличие API и CLI
6. Возможность настройки

Конвейер поиска повторов

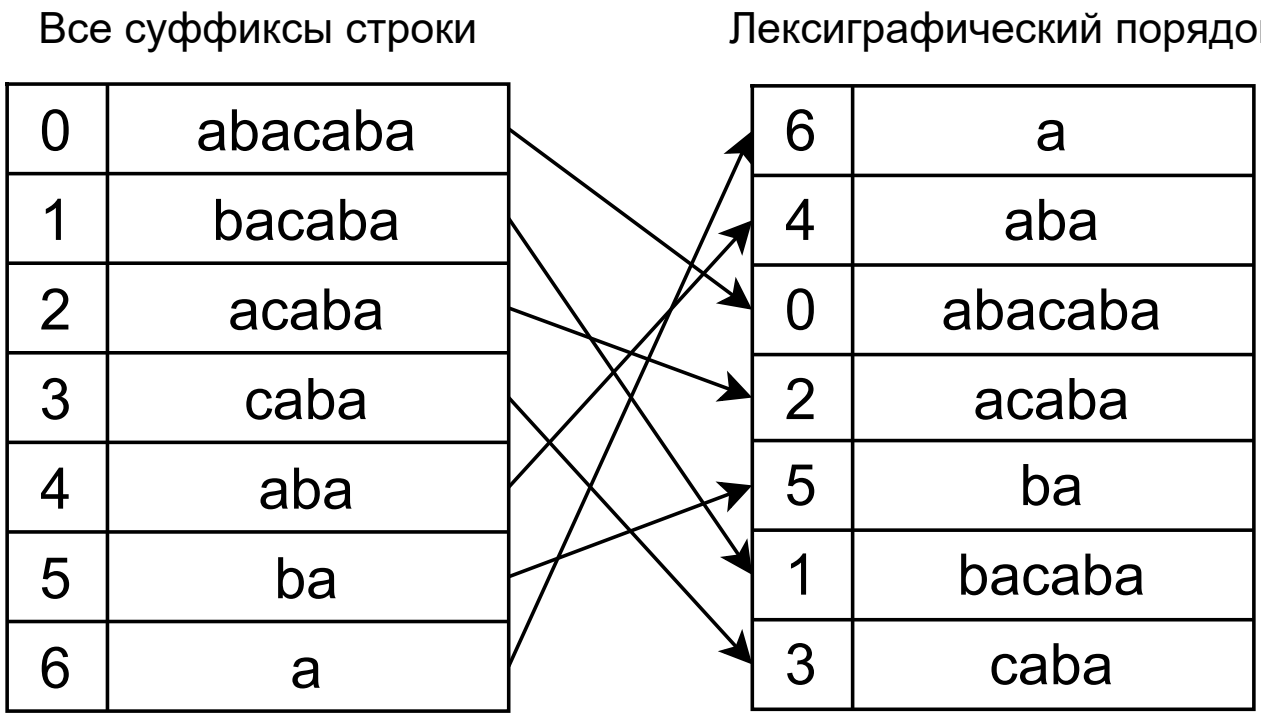


Предобработка текста

Методы NLP:

- Фильтрация спецсимволов
- Удаление стоп слов
- Лемматизация
- Стемминг

Поиск точных повторов

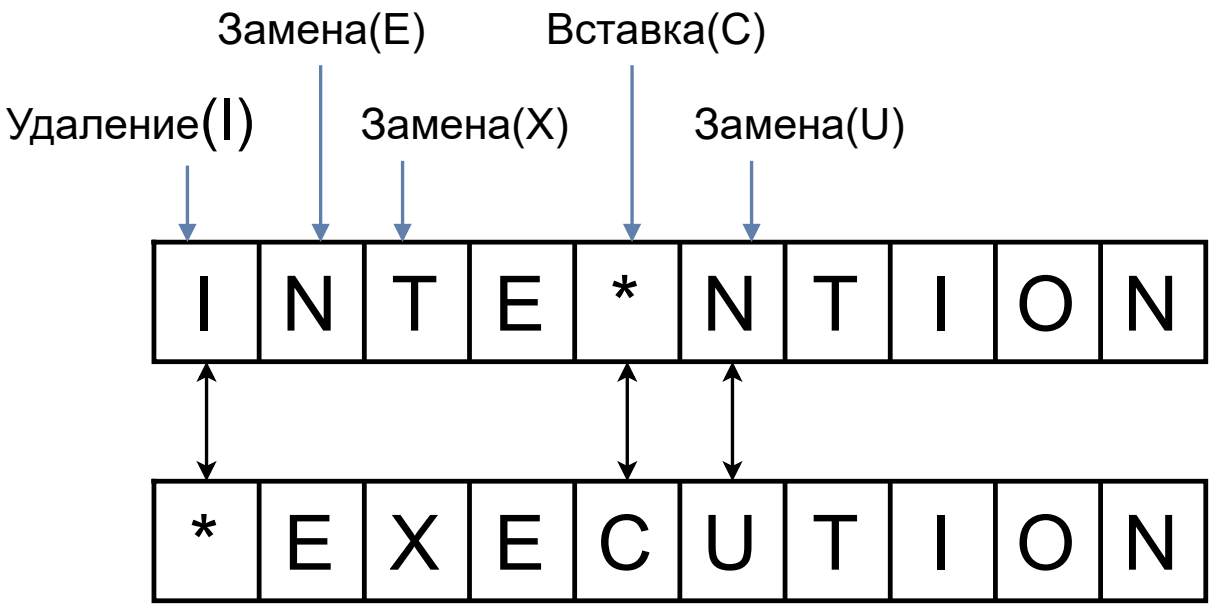


Суффиксный массив	6	4	0	2	5	1	3

LCP массив	-	1	3	1	0	2	0

Поиск неточных повторов I

Расстояние Левенштейна:



Поиск неточных повторов I

SimHash:

Hash 1	1	1	0	1	1	0	1	1
Hash 2	1	1	0	0	0	1	1	0
Hash 3	0	1	1	0	1	0	0	1
Result	1	1	0	0	1	0	1	1

Поиск неточных повторов II

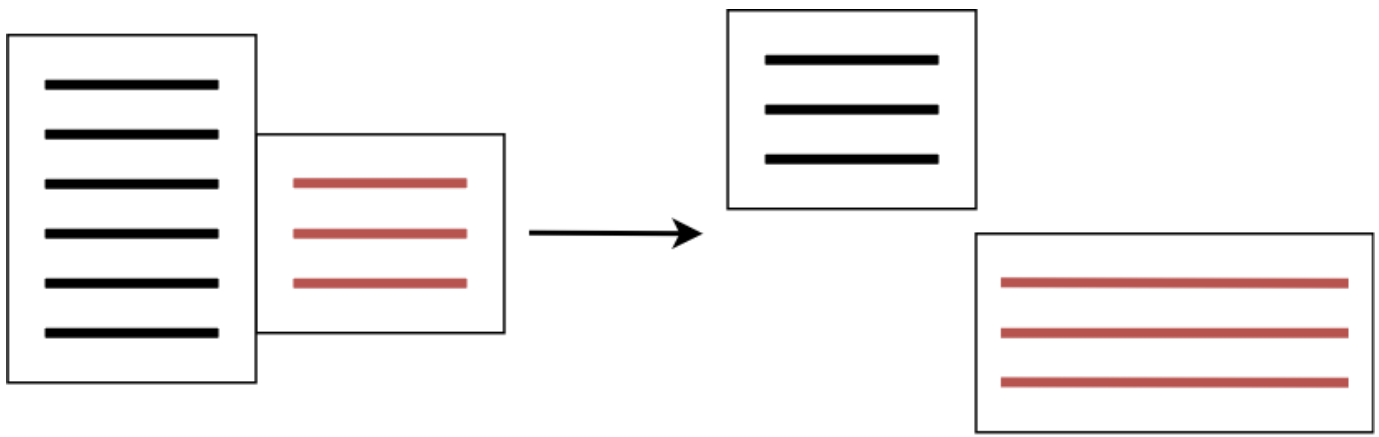
Множества N-грамм:

This is Big Data AI Book

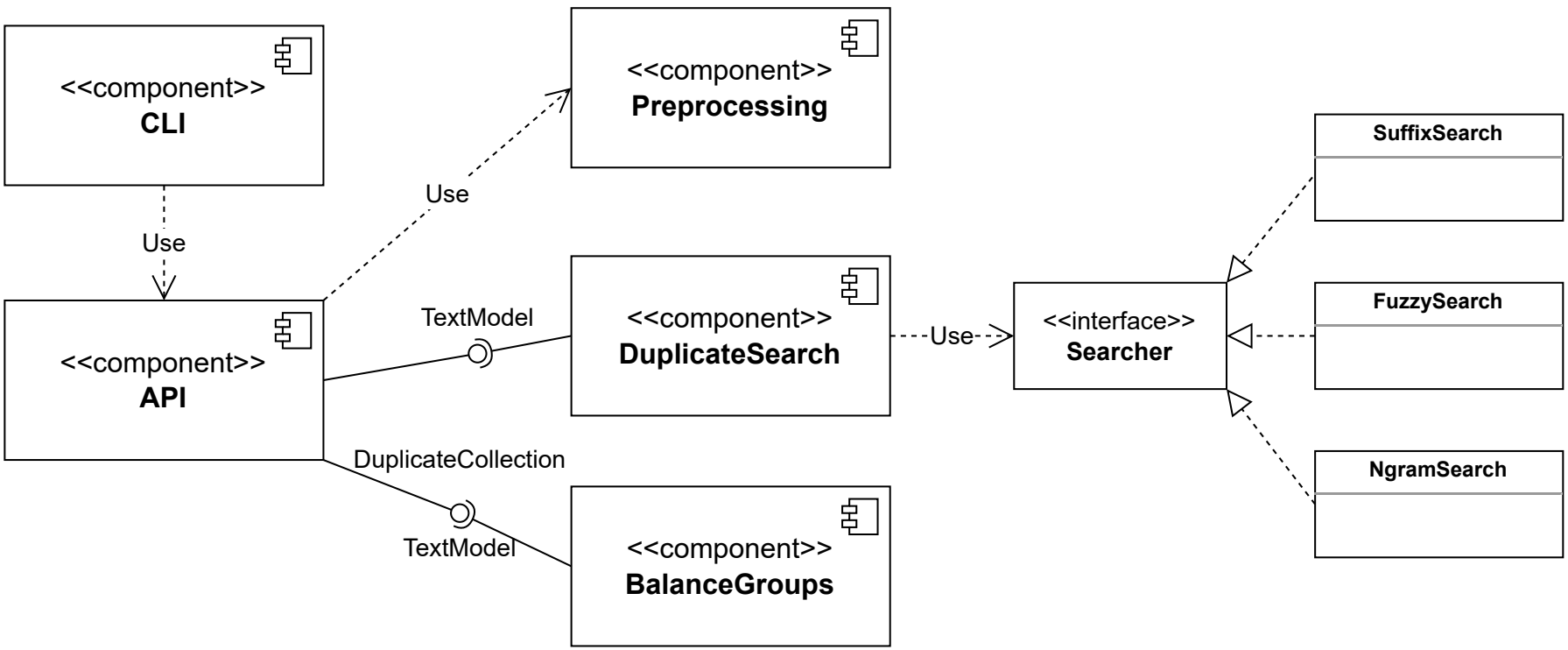
Uni-Gram	This	Is	Big	Data	AI	Book
Bi-Gram	This is	Is Big	Big Data	Data AI	AI Book	
Tri-Gram	This is Big	Is Big Data	Big Data AI	Data AI Book		

Балансировка групп повторов

- Фильтрация незначимых групп
- Слияние фрагментов



Архитектура



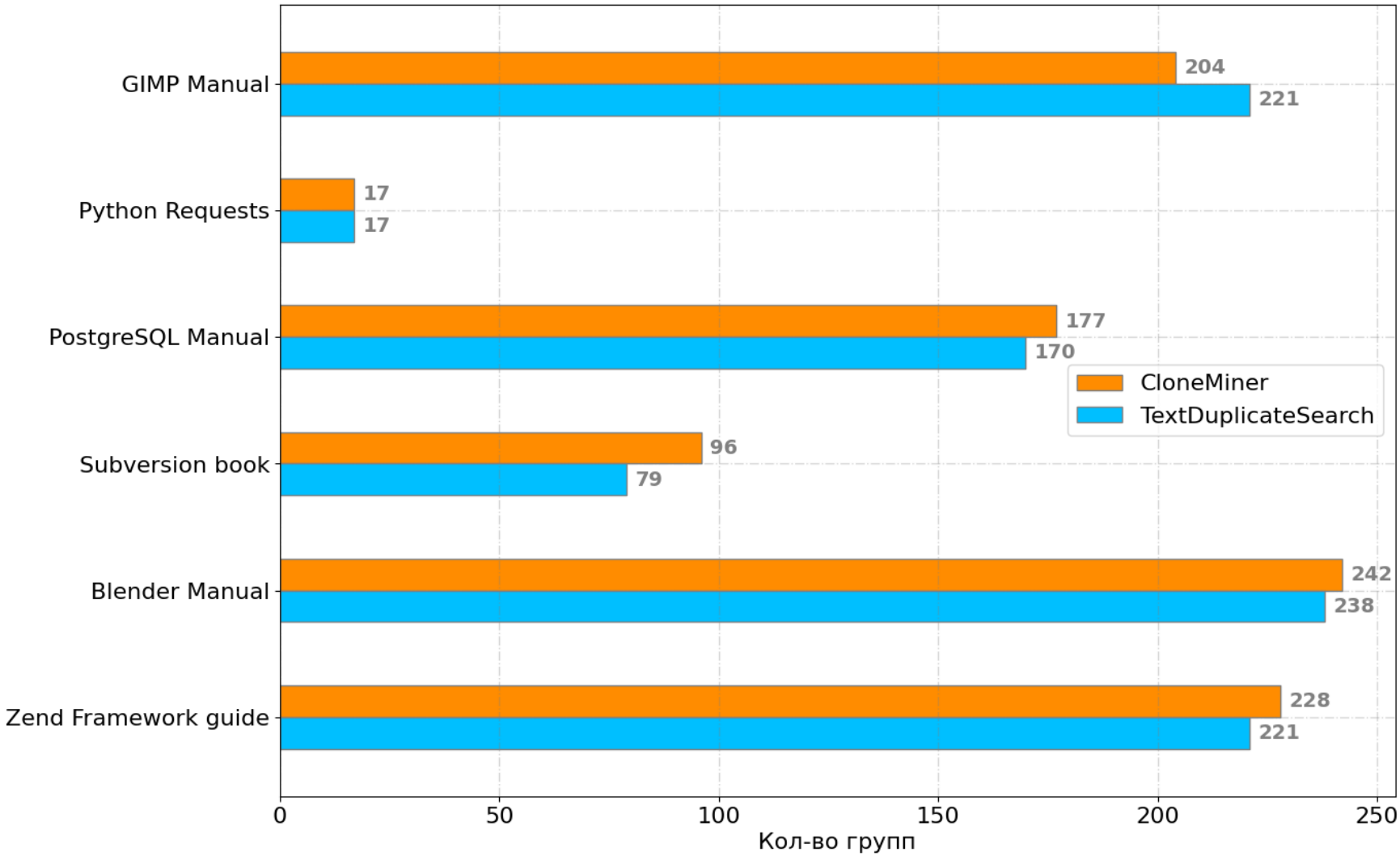
Тестирование: неточный поиск

Документ	Группы повто-ров	Средний размер группы	Средняя длина повтора	Покрытие документа
GIMP Manual	574	2,65	13,64	15%
PostgreSQL Manual	464	2,66	17,17	25%
Subversion book	282	2,27	18,93	10%
Zend Framework guide	522	2,32	22,96	16%
Blender Manual	1393	2,48	14,22	16%
Python Requests	23	2,26	20,16	28%

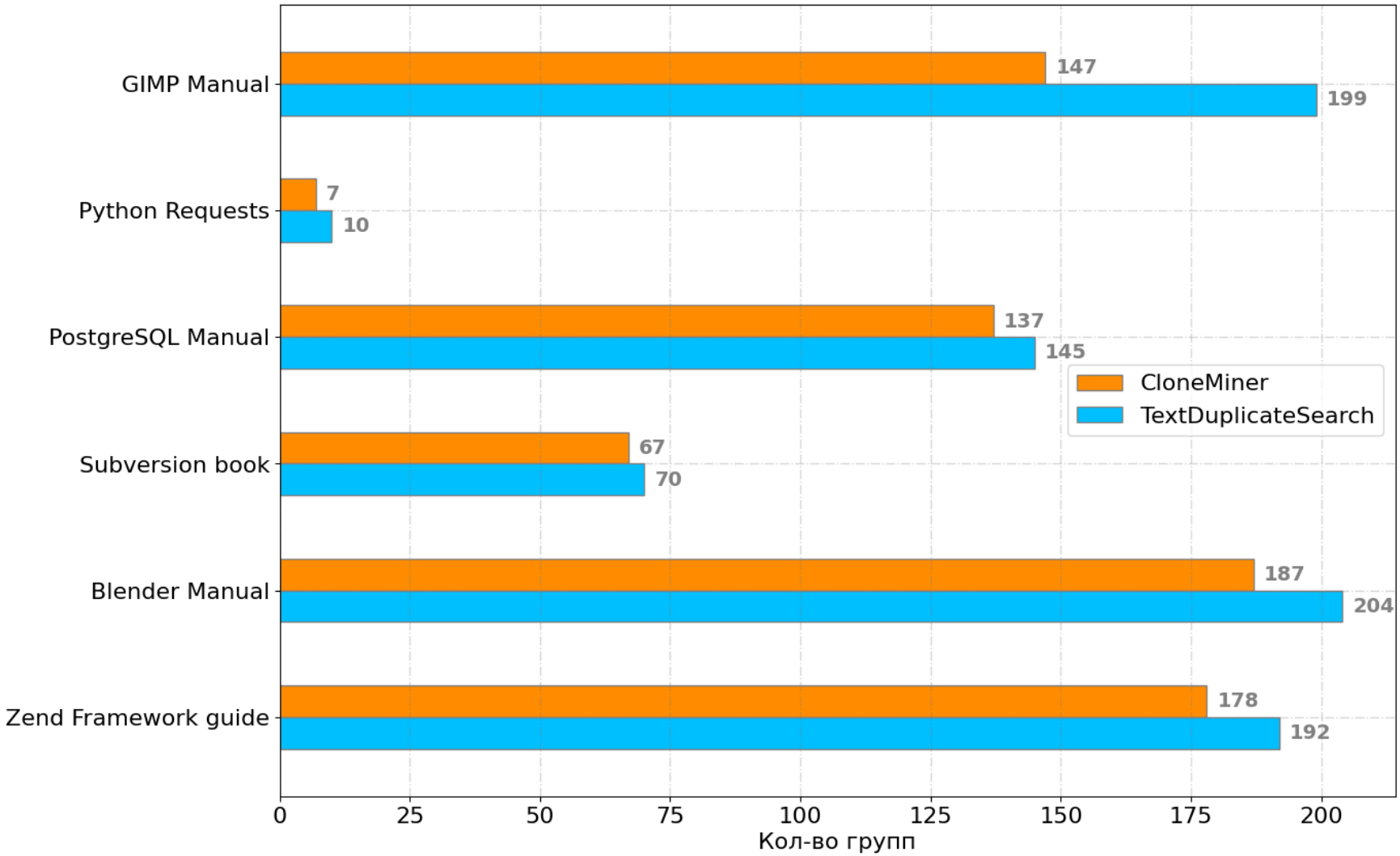
Тестирование: точный поиск

Документ	Группы повто-ров	Средний размер группы	Средняя длина повтора	Покрытие документа
GIMP Manual	400	2,57	14,59	11%
PostgreSQL Manual	289	2,30	16,31	14%
Subversion book	218	2,17	17,27	7%
Zend Framework guide	557	2,44	16,58	13%
Blender Manual	587	2,33	19,14	9%
Python Requests	24	2,58	18,83	31%

Тестирование: сравнение с CloneMiner



Тестирование: сравнение с CloneMiner



Тестирование: сравнение с CloneMiner

...

All keys are expected to be strings. The structure remembers the case of the last key to be set, and iter(instance), keys(), items(), iterkeys(), and iteritems() will contain case-sensitive keys. However, querying and contains testing is case insensitive:

...

All keys are expected to be strings. The structure remembers the case of the last key to be set, and iter(instance), keys(), items(), iterkeys(), and iteritems() will contain case-sensitive keys. However, querying and contains testing is case insensitive:

cid = CaseInsensawitiveDict() cid['Accept'] = 'application/json'

...

All keys are expected to be strings. The structure remembers the case of the last key to be set, and iter(instance), keys(), items(), iterkeys(), and iteritems() will contain case-sensitive keys. However, querying and contains testing is case insensitive:

...

All keys are expected to be strings. The structure remembers the case of the last key to be set, and iter(instance), keys(), items(), iterkeys(), and iteritems() will contain case-sensitive keys. However, querying and contains testing is case insensitive:

cid = CaseInsensawitiveDict() cid['Accept'] = 'application/json'

...

Тестирование: сравнение с CloneMiner

...

All keys are expected to be strings. The structure remembers the case of the last key to be set, and iter(instance), keys(), items(), iterkeys(), and iteritems() will contain case-sensitive keys. However, querying and contains testing is case insensitive:

...

All keys are expected to be strings. The structure remembers the case of the last key to be set, and iter(instance), keys(), items(), iterkeys(), and iteritems() will contain case-sensitive keys. However, querying and contains testing is case insensitive:
cid = CaseInsensawitiveDict() cid['Accept'] = 'application/json'

...

All keys are expected to be strings. The structure remembers the case of the last key to be set, and iter(instance), keys(), items(), iterkeys(), and iteritems() will contain case-sensitive keys. However, querying and contains testing is case insensitive:

...

All keys are expected to be strings. The structure remembers the case of the last key to be set, and iter(instance), keys(), items(), iterkeys(), and iteritems() will contain case-sensitive keys. However, querying and contains testing is case insensitive:
cid = CaseInsensawitiveDict() cid['Accept'] = 'application/json'

...

Заключение

1. Проанализированы основные подходы, используемые в существующих инструментах для поиска повторов
2. Выявлены требования к новому механизму поиска
3. Спроектирован конвейер для механизма поиска: предобработка текста, применение алгоритмов поиска повторов, балансировка групп повторов
4. Разработаны алгоритмы для точного и неточного поиска повторов на основе использованных в Duplicate Finder инструментов
5. Выполнена реализация инструмента на языке Python, исходный код выложен на GitHub; проведена интеграция с Duplicate Finder
6. Проведено тестирование инструмента на корпусе документов, по результатам работы собрана статистика и проведен ее анализ