УДК 004.912
DOI: 10.15827/0236-235X.134.257-268
Дата подачи статьи: 12.01.21
2021. Т. 34. № 2. С. 257–268

Алгоритмы автоматизации анализа текста на русском языке для решения прикладных задач с применением фреймворка TAWT

Е.В. Полицына [1], *к.т.н., доцент, kathrin.beaver@mail.ru*
С.А. Полицын [1], *к.т.н., доцент, pul_forever@mail.ru*
А.С. Поречный [1], *аспирант, alex.porechny@mail.ru*

[1] *Московский авиационный институт (национальный исследовательский университет), г. Москва, 125993, Россия*

В работе проведен обзор существующих инструментов лингвистического анализа текста. Выявлена проблема выбора подходящих инструментов, адаптации их для работы с текстами на русском языке и интеграции друг с другом. Именно эти аспекты затрудняют их применение в исследовательских целях и делают практически невозможным использование в прикладных системах. В настоящей статье описывается разработанный авторами уникальный Java-фреймворк TAWT с открытым исходным кодом, предоставляющий удобные готовые программные инструменты и структуры данных основных этапов анализа текста на русском языке, отвечающие современным требованиям к производительности, надежности, механизмам сборки проектов и т.д.

В статье предлагается подход к интеллектуализации информационных систем и бизнес-процессов с помощью программных средств лингвистического анализа текста для реализации алгоритмов автоматизации обработки технической документации, что составляет научную новизну работы. Применение разработанного фреймворка TAWT позволило реализовать алгоритмы автоматизации анализа текста на русском языке в части анализа технической документации: валидации структуры документов и перечня используемых в документе аббревиатур, поиска похожих документов и получения их краткого содержания. Все это упростит работу с технической документацией, ускорит процесс ее подготовки и повысит качество создаваемых документов.

Алгоритмы автоматического анализа текста, реализованные средствами фреймворка TAWT, были успешно применены в прикладных системах поиска друзей по интересам в социальных сетях, определения мошеннических сообщений, работы с тематическими синонимами, а также для создания программных средств выделения ключевых слов из текстов на русском языке и их реферирования.

Ключевые слова: *автоматизированный анализ текста, обработка естественного языка, компьютерная лингвистика, NLP, фреймворк для анализа текста, морфологический анализ, синтаксический анализ, семантико-синтаксический анализ, анализ текстовой документации.*

При решении самых разных задач в поисковых и новостных системах, системах документооборота и подготовки технической документации, системах электронной коммерции и др., которые кажутся далекими от лингвистического анализа текста, автоматизированный анализ текста может существенно упростить работу человека.

Разработчики компьютерных систем стремятся приблизить их функциональность к человеческим возможностям общения, распознавания и понимания естественного текста. Различные инструменты используются и в множестве коммерческих программных продуктов мониторинга СМИ (Interfax SCAN, продукты компании «Медиалогия»), отслеживания тенденций в какой-либо области, системах антиплагиата (Антиплагиат, Руконт) и т.д. Компания Яндекс также поддерживает работы в области автоматического реферирования веб-документов с учетом запроса [1].

Все это делает актуальным как разработку новых принципов и методов извлечения данных из текстов на естественном языке, так и создание на их основе новых информационных систем и интернет-технологий, включая средства поиска и анализа информации, приобретения знаний и создания онтологий. Интеллектуализация информационных систем и бизнес-процессов с помощью новых программных средств лингвистического анализа текста для реализации алгоритмов автоматизации обработки технической документации составляет научную новизну работы. Применение предложенного подхода демонстрируется на примере реализованных алгоритмов автоматизации ана-

лиза текста на русском языке для решения прикладных задач с использованием разработанного авторами фреймворка TAWT.

Для решения проблемы применения программных средств лингвистического анализа текстов в прикладных системах необходимы специализированные средства, поддерживающие популярные языки программирования для разработки промышленного ПО (Java, .Net и др.), которые ориентированы на решение прикладных задач и реализуют этапы лингвистической обработки.

За годы исследований было создано множество моделей, выдвинуто огромное количество гипотез и построенных на них методов, которые решают отдельные задачи анализа текста, причем часто с определенными ограничениями. Например, при снятии лексической и морфологической неоднозначности методами Synan, Trigra и Accopost точность может достигать более 90 % [2], а метод, реализованный в системе русско-английского и англо-русского фразеологического машинного перевода RETRANS, позволяет снять омонимию слов с 99 %-ной точностью, однако его ограничением является использование базового набора структур, расширение которого требует ручного поиска и добавления в систему [3, 4]. Предложены семантические модели с полной или частичной автоматизацией их построения, например, модель «Смысл–Текст» [5], применение концептуальных представлений для формализации семантической обработки текстов [6], подход к семантическому поиску в коллекции документов [7] и др. Проблема построения формальной модели естественного языка с возможностью ее автоматического построения и использования для выполнения прикладных задач не решена в полной мере, поэтому невозможно полное автоматическое понимание текста компьютерной программой без решения ряда отдельных задач компьютерной лингвистики.

Необходимость точной обработки естественного языка обусловливает создание разнообразных инструментов для анализа текста, таких как Lemmatizer, Greeb, Stemka, pyMyStem3, TreeTagger, Text Summarization, Tools4noobs и других, а также различных комплексов инструментов – AOT, GATE, LingPipe, UIMA, Texterra, а в коммерческом сегменте – Томита-парсер, SpeechKit и др. компании Яндекс, ядро Google, ABBYY. Такое разнообразие можно объяснить сложностью и отсутствием однозначного решения задач NLP (Natural Language Processing). Для создания и проверки новых подходов и гипотез необходимы средства разработки, позволяющие комплексно учитывать предыдущий опыт, накопленный в области компьютерной лингвистики, для создания на их базе новых инструментов.

В области автоматического анализа текстовой информации на русском языке существуют различные библиотеки и инструменты (Greeb, Lemmatizer, pymystem3, MaltParser). Инструменты могут быть как коммерческими (ABBYY, RCO, IBM, Яндекс), так и распространяемыми свободно (AOT, GATE). Однако найти программный продукт, ориентированный на разработку и при этом способный решать наиболее распространенные задачи NLP, за исключением некоторых API от Яндекс, Google, IBM, достаточно проблематично, так как существующие инструменты и библиотеки ориентированы на решение узконаправленных задач.

Анализ инструментов NLP

Существующие средства для автоматического анализа текста можно разбить на группы:

– лингвистические инфраструктуры с наличием фреймворков в их составе (GATE, LingPipe, Apache UIMA, Texterra);

– порталы с наборами сервисов анализа текста (META-SHARE, Lapps Grid, Language Grid, CLARIN);

– инструментальные средства, представляемые в виде программных интерфейсов (Google, Yandex, ABBYY, RCO, IBM Alchemy-API, Semantria и др.).

Наиболее популярные фреймворки (GATE, LingPipe, UIMA) разработаны для английского и других иностранных языков, при этом практически не имеют поддержки русского языка, что делает невозможным их использование для анализа документов на русском языке.

Попытки адаптировать фреймворки под русский язык могут занять значительное время, но так и не дать результата, поскольку многие методы и подходы, эффективные для одного языка, дают гораздо худшие результаты для другого. Например, алгоритм Портера эффективен для морфологического анализа текстов на английском языке, но для русского имеет низкую точность из-за особенностей языков. Аналогичные проблемы характерны и для алгоритмов синтаксического анализа, что связано с различиями в синтаксических правилах разных языков. Создание и развитие Texterra

подтверждает актуальность создания аналогичного фреймворка для русского языка, но еще более ориентированного на упрощение прикладного использования.

Существуют программные реализации каждого этапа анализа текста, однако такие решения являются разрозненными. Это означает, что для их совместного использования необходимо создавать интерфейсы между инструментами. Инструмент графематического анализа Greeb реализован на языке программирования Ruby, Twitter NLP and Part-of-Speech Tagger – Java. Инструменты морфологического анализа: MAnalyzer – C/C++, а также комплексы AOT – Python/C/C++/JavaScript. Совместное использование инструментов, реализованных на разных языках программирования, еще более увеличивает время и стоимость их интеграции, уменьшает производительность, так как требуется больше преобразований данных по сравнению с реализацией всех этапов анализа теста на одном языке программирования, может накладывать ограничения на поддерживаемые платформы и т.д, а также существенно усложняет последующее развертывание и сопровождение такого ПО.

Использование инструментов лингвистического анализа текста при разработке промышленных систем предполагает, что они будут внедряться в уже существующую систему или в систему, для разработки которой определен стек технологий. По данным исследований различных рейтинговых агентств, самым распространенным языком программирования в настоящее время является Java [8, 9].

Кроме того, разработчикам прикладных систем довольно сложно определить оптимальные структуры данных для хранения объектов для автоматического анализа текста, поскольку необходимо иметь солидный опыт в области компьютерной лингвистики и представление об особенностях естественно-языковых текстов.

Фреймворк TAWT

Реализация и использование библиотеки морфологического анализа текста JMorfSdk [10] подтвердили преимущества предоставления разработчикам ПО готовых структур данных для работы с лингвистическими конструкциями и существенно упростили разработку различных алгоритмов автоматического анализа текста тем, кто не специализируется в области лингвистики. Например, удалось улучшить обработку текстовых данных в приложении FriendsFinder для поиска людей по интересам в социальных сетях путем нормализации слова поискового запроса и данных со страниц пользователей, учета частей речи найденных словсовпадений и т.д., разработать новую версию мобильного приложения TouristHelper 2.0 с поддержкой русского языка, создать сервис подбора тематических синонимов [11], ускорить реализацию инструментов для проверки новых подходов к решению лингвистических задач, разметки корпусов текстов [12].

Для обеспечения возможности интеллектуализации информационных систем за счет автоматизации обработки текста с применением лингвистического анализа разных уровней в системах, в которых обработка текста является второстепенной задачей, авторами был разработан фреймворк TAWT (Tools for Automated Work with Text), состоящий из набора программных инструментов.

Структура и схема взаимодействия между отдельными инструментами представлены на рисунке 1. Все инструменты используют общую схему подключения, стандартную для платформы Java, – глобальный репозиторий бинарных зависимостей, исходный код которых, примеры и ссылки на артефакты находятся в общем доступе на Github [13].

Фреймворк разработан исходя из того, что предложение состоит из набора опорных оборотов. Это оборот, ограниченный запятыми или иными знаками препинания и обязательно содержащий хотя бы одно слово, являющееся опорной частью речи. Опорные части речи – глаголы, существительные, деепричастия и причастия, а слова, имеющие с ними словоформы, называются опорными словами [14].

Инструмент Parser реализует графематический этап анализа текста, основан на наборе правил, реализованных средствами регулярных выражений. В инструменте Parser используются различные наборы регулярных выражений для выделения опорных оборотов, предложений, абзацев для того, чтобы верно распознать нужные границы при анализе. Последующий разбор предполагает, что существенной разницы между сложными предложениями и набором опорных оборотов нет. На рисунке 2 приведен пример выделения предполагаемых опорных оборотов из предложений, одно из которых является сложносочиненным.

Итоговый набор опорных оборотов формируется инструментом GAMA после проведения морфологического анализа.

Инструмент JMorfSdk (Java Morphological Sdk) реализует морфологический этап ана-

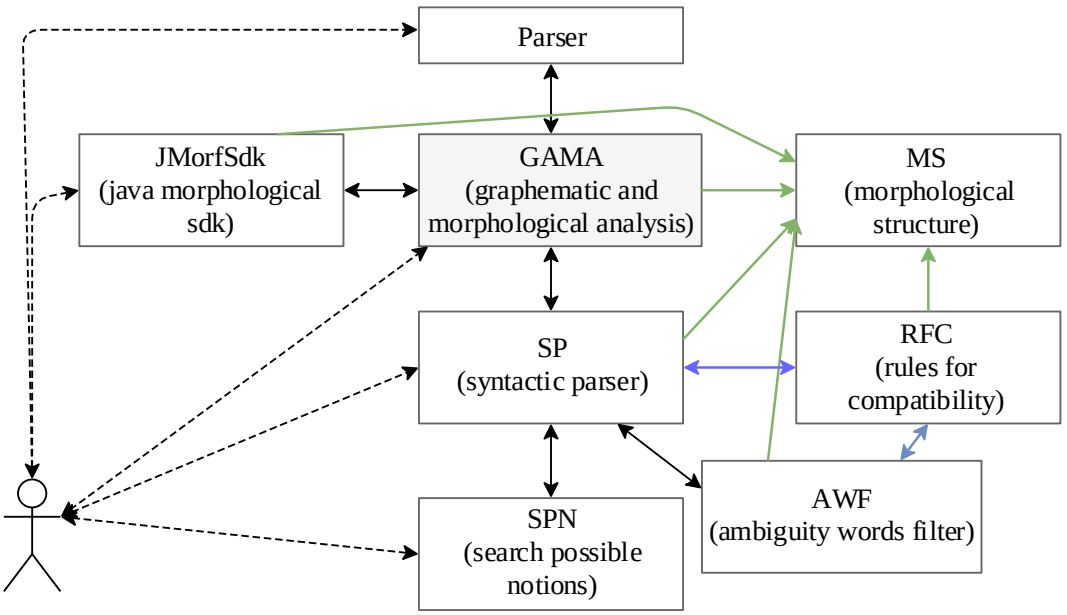

Рис. 1. Структура фреймворка TAWT

Fig. 1. TAWT Framework Structure

лиза текста, основан на модификации машиноориентированного «Грамматического словаря русского языка А.А. Зализняка», разработанного и поддерживаемого проектом OpenCorpora [15, 16]. На данный момент корпус содержит более 360 тысяч уникальных слов, более 5 млн словоформ, включающих редкие и новые слова, а также самые типичные опечатки для уменьшения их влияния на анализ текста в целом. JMorfSdk имеет высокую производительность, сложность получения множества омоформ слова и их морфологических характеристик составляет $O(1)$, что достигается за счет использования хэш-таблиц вместе с использованием битовых операций и хранением необходимых характеристик в битовой шкале. Пример фильтрации списка слов по женскому роду представлен на рисунке 3.

Средняя скорость выполнения морфологического анализа текста составляет 900 тысяч слов/с. Отличительной особенностью инструмента является наличие режима генерации слов по заданным морфологическим характеристикам [17].

MS (Morphological Structure) – инструмент загрузки, хранения, обработки и выдачи форм слова и его характеристик, содержит описание всех структур данных, конвертирует применяемые в JMorfSdk словари из исходного формата в формат, который используется во фреймворке.

Инструмент GAMA (Graphematic and Morphological Analysis) агрегирует методы графематического и морфологического этапов анализа текста, поддерживает замену инструментов графематического и морфологического анализа без перекомпиляции кода.

```
GraphematicParser parser = new GParserImpl();
List<List<String>> listBasicPhase = parser.parserSentence("Graphematic Parser - это программа"
        + " начального анализа естественно-языкового текста, рассматривающая его как цепочку"
        + " символов для получения информации, необходимой для следующих этапов анализа.");
System.out.println(listBasicPhase);
```

```
[
    [-, это, программа, начального, анализа, естественно-языкового, текста],
    [рассматривающая, его, как, цепочку, символов, для, получения, информации],
    [необходимой, для, следующих, этапов, анализа]
]
```

Рис. 2. Пример выделения предполагаемых опорных оборотов из предложений

Fig. 2. Extraction of the estimated bearing phrases from the sentences

```
List<String> words = Arrays.asList("осенний", "осенней", "площадь", "стол", "играть", "конференций", "на", "бежала");
for (String word : words) {
    jMorfSdk.getAllCharacteristicsOfForm(word).forEach(form -> {
        if (form.getTheMorfCharacteristics(MorfologyParameters.Gender.class) == MorfologyParameters.Gender.FEMININ) {
            System.out.println(form + " - " + word);
        }
    });
}
```

```
initialFormString = осенний, typeOfSpeech = 18, morfCharacteristics = 488 - осенней
initialFormString = площадь, typeOfSpeech = 17, morfCharacteristics = 555 - площадь
initialFormString = конференция, typeOfSpeech = 17, morfCharacteristics = 187 - конференций
initialFormString = бегу, typeOfSpeech = 20, morfCharacteristics = 670760 - бежала
```

Рис. 3. Пример фильтрации списка слов по женскому роду

Fig. 3. Filtering a list of words by the female gender

Инструмент RFC (Rules for Compatibility) предоставляет набор реализованных правил сочетаемости слов [18]. Включает в себя как простые правила, такие как сочетаемость предлога с существительным по падежу (причем предлог должен быть до существительного), согласование прилагательного и существительного по падежу, числу и роду и др., так и более сложные правила: глаголы, отглагольные прилагательные и наречия являются управляющими словами по отношению к существительному, если существительное стоит в именительном или винительном падеже и перед ним нет предлога, при этом расстояние между ними не должно быть более четырех слов и др.

Инструмент AWF (Ambiguity Words Filter) содержит фильтр устранения неоднозначности слов на основе правил из RFC, поддерживает замену инструмента RFC на другую модель, который подтверждает или отрицает сочетаемость пары входных слов. По результатам проверок снятия неоднозначности было установлено, что она выдает неправильный результат не более, чем в 15 % случаев [14].

Инструмент SP (Syntactic Parser) реализует семантико-синтаксический этап анализа, для его работы необходим предварительно обработанный текст, каждое слово которого имеет свои морфологические характеристики. Как отдельные компоненты используются фильтр устранения неоднозначности и набор правил сочетаемости слов для установления связей между словами в пределах опорного оборота, предложения и абзаца. На основе сформулированных правил разработаны алгоритмы определения опорных слов и установления связей между ними, так как неопорные слова в основном являются зависимыми для соседних опорных слов, причем чаще для правого, реже для левого опорного слова. Оставшиеся слова привязываются к стоящему справа опорному слову [18]. В результате работы алгоритма получается сеть или сети зависимостей, в корне которых находится сказуемое простого предложения или подлежащее, если сказуемого нет. На рисунке 4 приведен пример результата семантико-синтаксического анализа предложения со снятыми неоднозначностями.

Фильтр неоднозначности не всегда полностью снимает неоднозначность всех слов, например, при поиске связей между словами необходимо анализировать каждую комбинацию омоформ неоднозначных слов. Таким образом, результатом работы инструмента SP могут являться несколько сетей зависимостей.

SPN (Search Possible Notions) – инструмент поиска потенциальных понятий и ключевых словосочетаний на основе анализа сетей зависимостей, полученных в результате работы инструмента SP. Предполагается, что наиболее часто встречаемые словосочетания в большой выборке текстов, написанных человеком на естественном языке, могут являться понятиями [3].

Скорость выполнения семантико-синтаксического анализа зависит от стиля текста и в среднем составляет 22,2 предложения/с для публицистических и художественных текстов и 66,7 предложения/с для научных текстов. Вычислительная сложность алгоритма – $O(n)$, так как время работы алгоритмов фильтра и поиска связей в пределах опорного оборота близко к постоянному значению, а изменение в средней скорости работы инструмента при анализе одного предложения не зависит от объема текста.

Алгоритмы автоматизации анализа технической документации

Одна из областей, в которых целесообразно применение средств компьютерной лингви-

```
SyntaxParser sp = new SyntaxParser();
sp.init();
List<BearingPhraseSP> phrase
        = sp.getTreeSentence( text: "Стало ясно, что будет с российской валютой.");
phrase.forEach(System.out::println);
```

```
BearingPhraseSP{
  words=[
    word={стало,      ToS=20}, main=[нет],            dependents={ясно, ToS=9},
    word={ясно,       ToS=9},  main={стало,  ToS=20}, dependents=[нет]
  ],
  mainOmoForm={стало,     ToS=20}
}
BearingPhraseSP{
  words=[                                                                    ,
    word={будет,      ToS=20}, main=[нет],            dependents={с,           ToS=12},
    word={с,          ToS=12}, main={будет,  ToS=20}, dependents={валютой,    ToS=17}
    word={российской, ToS=18}, main={валютой, ToS=17}, dependents=[нет],
    word={валютой,    ToS=17}, main={с,      ToS=12}, dependents={российской, ToS=18}
  ],
  mainOmoForm={будет, ToS=20}
}
```

Рис. 4. Пример результата семантико-синтаксического анализа предложения со снятыми неоднозначностями

Fig. 4. The result of semantic-syntactic analysis of a sentence with clarified words

стики, – системы документооборота и в целом подготовка разного рода документации: проектной, конструкторской и др. В большинстве случаев к документам предъявляются требования по структуре, оформлению, содержанию. Подготовка и обновление документации, составление пакетов документов при создании и развитии ПО, технических решений и т.д. – очень трудоемкая задача.

Применение инструментов фреймворка TAWT разных уровней позволяет существенно упростить многие задачи подготовки технической документации. В таблице приведены некоторые решаемые прикладные задачи в области анализа текстовой документации и используемые инструменты фреймворка.

Валидация структуры документа

Техническая документация имеет структуру, которая определяется одним из стандартов, например, для ТЗ это ГОСТ 34.602-89, ГОСТ 19.201-78, IEEE STD 830-1998 и т.д. Валидация структуры документа в соответствии со стандартами актуальна как при обучении новых сотрудников, так и для упрощения работы опытных специалистов, поскольку позволяет в автоматизированном режиме делать первичную проверку документа, а также повышает качество разработки технической документации.

Реализованный алгоритм валидации использует инструмент Parser, а также Apache POI – библиотеку для работы с файлами формата .docx.

Для оценки качества работы алгоритма валидации были проанализированы на соответствие ГОСТ 34.602-89 ТЗ из открытой базы на разработку и/или модернизацию ПО для Минэкономразвития РФ. В результате выделены наиболее распространенные отклонения структуры документов от ГОСТа: неверный уровень разделов и подразделов, изменение названий разделов вместо использования подразделов.

Валидация сокращений в списке терминов и сокращений

Техническая документация содержит большое количество различных специфичных терминов и сокращений, которые могут не являться общеупотребимыми или иметь неочевидную расшифровку, зависящую от предметной области. Для однозначного определения такие термины и сокращения выносят в отдельный раздел документа «Термины и сокращения».

Однако техническая документация может создаваться продолжительное время, поэтому ее разработчик может забыть указать все сокращения в соответствующий раздел или, наобо-

Инструменты комплекса для решения прикладных задач

Used tools of the complex for solving applied problems

Задача	Используемые инструменты фреймворка TAWT
Валидация структуры документа на соответствие ГОСТу	Parser, JMorfSdk, MS
Валидация наличия в разделе «Термины и определения» всех употребляемых в тексте документа аббревиатур	Parser, JMorfSdk, MS
Поиск схожих технических решений на основе ТЗ или описания эксплуатационных условий	Parser, JMorfSdk, GAMA, SP, AWF, RFC, MS, SPN
Построение краткого содержания документа	Parser, JMorfSdk, MS

рот, внести лишние сокращения, которые в тексте не используются. Постоянная актуализация документов трудоемка и часто приводит к появлению множества ошибок. Тогда возникает необходимость автоматически получать список аббревиатур и сокращений, используемых в тексте, а также проверять их наличие в разделе «Термины и сокращения».

Реализация алгоритма валидации с использованием графематического анализа без использования морфологического анализа текста дает неточный результат ввиду того, что невозможно на графематическом уровне отличить слово, написанное заглавными буквами, от аббревиатуры.

Результаты поиска аббревиатур несколькими способами показали следующее.

- Необходимо использовать графематический анализ, который учитывает все возможные конструкции приложений, в том числе все символы разделения и т.д., или применять дополнительную последующую обработку для объединения выделенных аббревиатур.
- Аббревиатура обязательно пишется заглавными буквами, но этого недостаточно, чтобы определить слово как аббревиатуру. Так, без использования морфологического анализа слова «ТЕХНОЛОГИЯ», «ПРИНЦИПЫ» и т.д. будут определены как аббревиатура. Для уточнения, является ли данное слово аббревиатурой в рассматриваемом документе, необходимо проводить морфологический анализ слов и определять, что морфологически характеристики слова не содержат признак аббревиатуры, а также учитывать имеющийся список терминов и аббревиатур документа.

Поиск похожих документов

Задача поиска схожих по смыслу документов является актуальной в случае, если имеется большая база технической документации и необходимо по исходному документу, например ТЗ, найти уже разработанные технические решения, которые упростят разработку нового продукта или изделия.

Для оценки результатов поиска похожих документов средствами фреймворка TAWT были реализованы несколько методов поиска.

- Статистический поиск по словам средствами инструмента Parser без применения морфологического анализа. Особенностью является то, что метод реализуется быстро, однако имеет очевидные недостатки, слова в разных формах будут считаться уникальными, например, слова «системы» и «систему» уникальные.
- Статистический поиск по словам с применением морфологической обработки включает в себя использование инструмента GAMA. Применение морфологического анализа позволяет приводить каждое слово к начальной форме, таким образом слова «системы» и «систему» будут приведены к «системе» и считаться одним уникальным словом.
- Статистический поиск по ключевым словам. Особенностью поиска по ключевым словам является то, что исключаются остальные слова, которые не несут смысловой нагрузки.
- Статистический поиск по ключевым словосочетаниям. Особенностью данного поиска является то, что ключевым элементом считается словосочетание, несущее более точную смысловую нагрузку, чем ключевое слово. Для этого используется инструмент SPN, который устанавливает синтаксические связи между словами и выделяет ключевые словосочетания.

Для оценки подходов использовались ТЗ из области автоматизации ПО, разработки прикладного ПО, конструирования деталей, а также документы, не относящиеся к ТЗ, такие как отчеты, описания и рецензии.

Анализ результатов, полученных авторами, показал, что статистический поиск по словам без морфологической обработки, с морфологической обработкой и по ключевым словам дает примерно одинаковый результат, однако поиск по ключевым словосочетаниям отсеивает документы, которые не подходят по содержанию, и точнее отделяет схожие документы от возможно схожих, тем самым уменьшая время, необходимое на последующий ручной анализ.

Выбранный критерий (соотношение количества найденных элементов и элементов в исходном тексте) не является оптимальным, так как если по размеру документ будет многократно превышать исходный, то количество совпадающих элементов может быть большим, но не из-за схожести, а из-за разницы в объеме. Поэтому данный критерий подходит для обработки примерно равных по объему документов, в других случаях необходимо использовать «веса» в зависимости от особенностей анализируемых документов.

Получение краткого содержания документа

Помимо автоматической фильтрации не подходящих для решаемой задачи документов, еще одним способом ускорения поиска и анализа нужной технической документации является сокращение объема текста документа без существенной потери смысла. Для этого применяются методы аннотирования и реферирования.

С применением инструмента JMorfSdk фреймворка для морфологического анализа текста была разработана библиотека, содержащая ряд различных методов реферирования, наиболее оптимальными являются интегральный метод [19] и сервис автоматического реферирования текста [20] на ее основе.

Для упрощения поиска документов, схожих с ТЗ на разработку новых модулей ИС, были получены рефераты по автоматически отобранным документам. Например, ТЗ, изложенное на 90 страницах, содержит общие сведения, цели доработки, характеристики объекта автоматизации, требования к системе и к вводу доработок в эксплуатацию. Все перечисленные пункты имеют отображение в полученном реферате объемом 7 страниц, что дает возможность человеку гораздо быстрее определить, подходит ли он для дальнейшей работы, в какие разделы нужно внести изменения, сравнить и проверить данные и т.д.

Таким образом, разработанные с применением фреймворка TAWT алгоритмы автоматизации подготовки технической документации демонстрируют возможности упрощения и ускорения работы технических писателей, аналитиков и других специалистов.

Применение фреймворка TAWT для решения прикладных задач

Основными направлениями автоматизации обработки текстовых технических документов с применением фреймворка TAWT являются автоматизация решения отдельных практических задач с применением инструментов компьютерной лингвистики и с учетом особенностей документов в определенной области, компании и пр. и разработка комплексных решений на основе применения отдельных инструментов анализа текстовой документации.

Отдельные инструменты разработанного авторами фреймворка TAWT успешно применялись для реализации алгоритмов автоматического анализа текстов при создании исследовательских программных средств. Все разработанные инструменты показали хорошее качество распознавания именованных сущностей (более 90 %), правильную генерацию слов при расшифровке сокращений более чем в 70 % случаев и возможность дальнейшего совершенствования алгоритмов расшифровки сокращений, улучшение качества выделения ключевых слов на 8 % относительно известных методов [21], реализацию интегрального метода реферирования с улучшением качества построения реферата в среднем на 15 % [19]. Все разработанные инструменты и библиотеки доступны на портале «Автоматизированный анализ текста» [20].

Инструменты MS и JMorfSdk применяются для предобработки текстов в ансамбле классификаторов на основе машинного обучения и нейронных сетей [22], для получения морфологической разметки корпуса текстов [23], для вторичной обработки сообщений социальных сетей в плагине браузера Google Chrome для обнаружения мошеннических сообщений с точностью распознавания 97,5 % [24].

Заключение

Разработанный авторами фреймворк является средством для реализации алгоритмов лингвистического анализа текста разных уровней, решения прикладных задач и быстрой про-

верки различных гипотез в компьютерной лингвистике [25]. TAWT – кроссплатформенное решение с открытым исходным кодом, ориентированное на быстрое внедрение и использование.

Реализация алгоритмов лингвистического анализа позволяет повысить качество обработки текстовых данных во многих областях и открывает новые возможности для автоматизации обработки текстовых документов, используя накопленный опыт в области компьютерной лингвистики и особенности решаемых задач. На примере валидации структуры документа и списка используемых аббревиатур продемонстрировано преимущество применения средств графематического и морфологического анализа, а поиск похожих документов и сокращение объема найденных документов существенно ускоряют процесс их анализа и упрощают работу с большим набором документов.

Фреймворк поддерживается и развивается, он использовался при создании других приложений автоматического анализа текста и прикладных систем: сервисов реферирования текстов, поиска людей по интересам в социальных сетях, подбора тематических синонимов и др.

Предложенные алгоритмы автоматизации анализа технической документации позволят разработчикам систем документооборота и других систем работы с текстовой документацией сократить рутинные действия, ускорить и упростить работу пользователей.

Литература

1. Браславский П., Колычев И. Автоматическое реферирование веб-документов с учетом запроса. В кн.: Интернет-математика. Автоматическая обработка веб-данных. М., 2005. С. 485–501.

2. Сокирко А.В., Толдова С.Ю. Сравнение эффективности двух методик снятия лексической и морфологической неоднозначности для русского языка. В кн.: Интернет-математика 2005. Автоматическая обработка веб-данных. М., 2005. С. 80–94.

3. Белоногов Г.Г., Зеленков Ю.Г., Кузнецов Б.А., Новоселов А.П. и др. Автоматизация составления и ведения словарей для систем фразеологического машинного перевода текстов с русского языка на английский и с английского на русский // НТИ. Сер. 2. Информационные процессы и системы. 1993. № 12. С. 16–21.

4. Белоногов Г.Г., Зеленков Ю.Г., Кузнецов Б.А., Новоселов А.П., Пащенко Н.А., Хорошилов Александр А., Хорошилов Алексей А. Интерактивная система русско-английского и англо-русского машинного перевода политематических научно-технических текстов // НТИ. Сер. 2. Информационные процессы и системы. 1993. № 3. С. 20–27.

5. Большаков И.А., Гельбух А.Ф. Модель «Смысл ⇔ Текст»: Тридцать лет спустя. IFID, 2000, № 1 (рус.). URL: https://www.gelbukh.com/CV/Publications/2000/Forum-MTM-rus.htm (дата обращения: 23.12.2020).

6. Разоренов А.А., Фомичев В.А. Новый подход к формализации семантической обработки предписаний на основе теории К-представлений // Информационные технологии. 2017. Т. 23. № 1. С. 3–14.

7. Chirkova N.A., Vorontsov K.V. Additive regularization for hierarchical multimodal topic modeling. Machine Learning and Data Analysis, 2016, vol. 2, no. 2, pp. 187–200. DOI: 10.21469/22233792.2.2.05.

8. PYPL PopularitY of Programming Language. URL: http://pypl.github.io/PYPL.html (дата обращения: 23.12.2020).

9. Tiobe – the Software Quality Company. URL: https://www.tiobe.com/tiobe-index (дата обращения: 23.12.2020).

10. Politsyna E.V., Politsyn S.A., Porechny A.S. Development of the cross-platform library of morphological analysis of the russian language text for industrial software. Proc. XIV CEE-SECR, 2018, art. 11, pp. 1–8. DOI: 10.1145/3290621.3290635.

11. Сервис тематических синонимов. URL: http://boberpul2.asuscomm.com:8088 (дата обращения: 23.12.2020).

12. Попов С.С., Полицын С.А., Полицына Е.В. Разработка комплекса инструментов для управления корпусами текстов // Информатика: проблемы, методология, технологии: матер. XIX Междунар. науч.-методич. конф. 2019. С. 1621–1626.

13. TAWT Framework and Tools Sources. URL: https://github.com/jalexpr (дата обращения: 23.12.2020).

14. Белоногов Г.Г. Теоретические проблемы информатики. Т. 2. Семантические проблемы информатики. М., 2008. 223 с.

15. Поречный А.С. Реализация кроссплатформенного фреймворка для автоматического анализа естественно-языкового текста // Гагаринские чтения: сб. тр. XLIV Междунар. конф. 2018. С. 112–113.

16. OpenCorpora. URL: http://opencorpora.org (дата обращения: 23.12.2020).

17. Поречный А.С., Полицына Е.В., Полицын С.А. Создание кроссплатформенной библиотеки морфологического анализа для русского языка // Информатика: проблемы, методология, технологии: матер. XVIII Междунар. науч.-методич. конф. Воронеж. 2018. Т. 7. С. 82–87.

18. Поречный А.С. Реализация поиска понятий с помощью выделение словосочетаний из текста // Проблемы компьютерной лингвистики и типологии. 2017. Вып. 6. С. 108–118.

19. Касаткина А.О. Разработка интегрального метода автоматического реферирования текста // Гагаринские чтения: сб. тр. XLIV Междунар. конф. 2019. С. 355–356.

20. Сервис автоматического реферирования текста. URL: http://textanalysis.ru/ (дата обращения: 23.12.2020).

21. Иващенко М.В. Анализ методов автоматизированного выделения ключевых слов из текстов на естественном языке // Информатика: проблемы, методология, технологии: матер. XIX Междунар. науч.-методич. конф. 2018. Т. 6. С. 19–24.

22. Самохин А.А. Анализ алгоритмов и расширение функциональности инструментов устранения сокращений в текстах на русском языке // Информатика: проблемы, методология, технологии: матер. XIX Междунар. науч.-методич. конф. 2019. Т. 1. С. 1627–1632.

23. Попов С.С., Полицын С.А., Полицына Е.В. Разработка комплекса инструментов для управления корпусами текстов // Информатика: проблемы, методология, технологии: матер. XIX Междунар. науч.-методич. конф. 2019. Т. 1. С. 1621–1626.

24. Петраш Я.П., Тихонова Д.А. Разработка инструмента для выявления мошеннических сообщений социальной сети «ВКонтакте» // Информатика: проблемы, методология, технологии: матер. XIX Междунар. науч.-методич. конф. 2019. С. 1513–1518.

25. Politsyna E.V., Politsyn S.A., Porechny A.S. The framework for hypothesis verification and analysis of natural language processing for the Russian language. Proc. VII Intern. Conf. AIST-SUP, 2018, vol. 2268, pp. 25–33.

Software & Systems
DOI: 10.15827/0236-235X.134.257-268

Received 12.01.21
2021, vol. 34, no. 2, pp. 257–268

Algorithms of the automatic text analysis for the Russian language for solving applied problems using TAWT framework

E.V. Politsyna [1], *Ph.D. (Engineering), Associate Professor, kathrin.beaver@mail.ru*
S.A. Politsyn [1], *Ph.D. (Engineering), Associate Professor, pul_forever@mail.ru*
S.A. Porechny [1], *Postgraduate Student, alex.porechny@mail.ru*

[1] *Moscow Aviation Institute (National Research University), Moscow, 125993, Russian Federation*

Abstract. The paper reviews the existing tools of linguistic text analysis. The authors identified the problem of selecting suitable tools, adapting them to work with texts in Russian, and integrating them with each other. This makes it difficult to use these tools both for research purposes and makes it almost impossible to use them in applied systems. The paper describes the new open-source Java framework TAWT, developed by the authors, that provides convenient ready-made software tools and data structures for the main stages of text analysis in the Russian language that meet the modern requirements for performance, reliability, project build engine, etc.

The paper proposes an approach to the intellectualization of information systems and business processes using software tools for linguistic text analysis to implement algorithms for automating the processing of technical documentation, which is the scientific novelty of the work. The application of the developed TAWT framework allowed implementing algorithms for automating the analysis of the text in Russian in terms of the analysis of technical documentation: validating the structure of documents and the list of abbreviations used in the document, searching for similar documents, and obtaining their brief content. All this will simplify the technical documentation management, speed up the process of its preparation, and upgrade the quality of the created documents.

The algorithms of automatic text analysis implemented by the TAWT framework have been successfully applied in application systems for searching friends by interests in social networks, identifying fraudulent mes-

sages, working with thematic synonyms, as well as for creating software tools for selecting keywords from texts in Russian and referencing them.

Keywords: automated text analysis, natural language processing, computational linguistics, NLP, text analysis framework, morphological analysis, syntax analysis, semantic-syntactic analysis, text documentation analysis.

References

1. Braslavsky P.I., Kolychev I.S. Query-biased summarization of web documents. In: *Internet Mathematics. Automatic Processing of Web Data*, Moscow, 2005, pp. 485–501 (in Russ.).
2. Sokirko A.V., Toldova S.Yu. Comparison of the effectiveness of two methods of removing lexical and morphological ambiguity for the Russian language. In: *Internet Mathematics 2005. Automatic Processing of Web Data*, Moscow, 2005, pp. 80–94 (in Russ.).
3. Belonogov G.G., Zelenkov Yu.G., Kuznetsov B.A., Novoselov A.P., Khoroshilov Alexander A., Khoroshilov Alexey A. Automation of collecting and maintaining dictionaries for systems of phraseological computer translation of texts from Russian into English and from English to Russian. *Nauchno-Tehnicheskaja Informacija. Ser. 2. Informacionnye Processy i Sistemy*, 1993, no. 12, pp. 16–21 (in Russ.).
4. Belonogov G.G., Zelenkov Ju.G., Kuznetsov B.A., Novoselov A.P., Pashchenko N.A., Khoroshilov Aleksandr A., Khoroshilov Aleksey A. An interactive system of Russian-English and English-Russian machine translation of polythematic scientific and technical texts. *Nauchno-Tehnicheskaja Informacija. Ser. 2. Informacionnye Processy i Sistemy*, 1993, no. 3, pp. 20–27 (in Russ.).
5. Bolshakov I.A., Gelbukh A.F. The Meaning⇔Text Model: Thirty years after. *IFID Publ*., 2000, no. 1. Available at: https://www.gelbukh.com/CV/Publications/2000/Forum-MTM-eng.htm (accessed December 23, 2020).
6. Razorenov A.A., Fomichev V.A. A new approach to formalization of instructions' semantic processing based on the theory of K-representations. *Information Technologies*, 2017, vol. 23, no. 1, pp. 3–14 (in Russ.).
7. Chirkova N.A., Vorontsov K.V. Additive regularization for hierarchical multimodal topic modeling. *Machine Learning and Data Analysis*, 2016, vol. 2, no. 2, pp. 187–200. DOI: 10.21469/22233792.2.2.05.
8. *PYPL PopularitY of Programming Language*. Available at: http://pypl.github.io/PYPL.html (accessed December 23, 2020).
9. *Tiobe – the Software Quality Company*. Available at: https://www.tiobe.com/tiobe-index (accessed December 23, 2020).
10. Politsyna E.V., Politsyn S.A., Porechny A.S. Development of the cross-platform library of morphological analysis of the russian language text for industrial software. *Proc. XIV CEE-SECR*, 2018, art. 11, pp. 1–8. DOI: 10.1145/3290621.3290635.
11. *Service of Thematic Synonyms*. Available at: http://boberpul2.asuscomm.com:8088 (accessed December 23, 2020).
12. Popov S.S., Politsyn S.A., Politsyna E.V. The development of the toolset for text corpuses management. *Proc. XIX Intern. Conf. Informatics: Problems, Methods, Technologies*, 2019, vol. 1, pp. 1621–1626 (in Russ.).
13. *TAWT Framework and Tools Sources*. Available at: https://github.com/jalexpr (accessed December 23, 2020).
14. Belonogov G.G. *Theoretical Problems of Computer Science. Vol. 2. Semantic Problems of Computer Science*. Moscow, 2008, 223 p. (in Russ.).
15. Porechny A.S. Implementation of cross-platform framework for automatic analysis of natural language text. *Proc. XLIV Gagarin Science Conf*. Russia, Moscow, 2018, pp. 112–113 (in Russ.).
16. *OpenCorpora*. Available at: http://opencorpora.org (accessed December 23, 2020).
17. Porechny A.S., Politsyna E.V., Politsyn S.A. The development of the cross-platform library for morphological analysis of texts in the Russian language. *Proc. XVIII Int. Conf. Informatika: Problemy, Metodologiya, Tekhnologii*. Russia, Voronezh, 2018, vol. 7, pp. 82–87 (in Russ.).
18. Porechny A.S. Implementation of the search for notions by selecting word combinations from text. *Problemy Komputernoy Lingvistiki i Tipologii*, 2017, no. 6, pp. 108–118 (in Russ.).
19. Kasatkina A.O. Development of a joint method for automatic text summarization. *Proc. XLIV Gagarin Science Conf*. Russia, Moscow, 2019, pp. 355–356 (in Russ.).
20. *Automatic Text Abstracting Service*. Available at: http://textanalysis.ru/ (accessed December 23, 2020).
21. Ivashchenko M.V. Analysis of automated keyword extraction methods in natural language texts. *Proc. XIX Intern. Conf. Informatika: Problemy, Metodologiya, Tekhnologii*, 2018, vol. 6, pp. 19–24 (in Russ.).

22. Samokhin A.A. Analysis of algorithms and the expansion of the functionality of tools to eliminate abbreviations in texts in Russian. *Proc. XIX Intern. Conf. Informatika: Problemy, Metodologiya, Tekhnologii*, 2019, vol. 1, pp. 1627–1632 (in Russ.).

23. Popov S.S., Politsyn S.A., Politsyna E.V. The development of the toolset for text corpuses management. *Proc. XIX Intern. Conf. Informatika: Problemy, Metodologiya, Tekhnologii*, 2019, vol. 1, pp. 1621–1626 (in Russ.).

24. Petrash Yu.P., Tikhonova D.A. Development of the tool for identification of fraud messages in social network “VKontakte”. *Proc. XIX Intern. Conf. Informatika: Problemy, Metodologiya, Tekhnologii*, 2019, vol. 1, pp. 1513–1518 (in Russ.).

25. Politsyna E.V., Politsyn S.A., Porechny A.S. The framework for hypothesis verification and Analysis of natural language processing for the Russian language. *Proc. VII Intern. Conf. AIST-SUP*, 2018, vol. 2268, pp. 25–33.

Для цитирования

Полицына Е.В., Полицын С.А., Поречный А.С. Алгоритмы автоматизации анализа текста на русском языке для решения прикладных задач с применением фреймворка TAWT // Программные продукты и системы. 2021. Т. 34. № 2. С. 257–268. DOI: 10.15827/0236-235X.134.257-268.

For citation

Politsyna E.V., Politsyn S.A., Porechny S.A. Algorithms of the automatic text analysis for the Russian language for solving applied problems using TAWT framework. *Software & Systems*, 2021, vol. 34, no. 2, pp. 257–268 (in Russ.). DOI: 10.15827/0236-235X.134.257-268.