Solving practical tasks of computer linguistics using the created text processing framework

Politsyna E V, Politsyn S A and Porechny A S

Branch of Intelligent monitoring systems Department #3, Moscow Aviation Institute (National Research University), 4, Volokolamskoye sh., Moscow, 125993, Russia

E-mail: kathrin.beaver@mail.ru

Abstract. The use of linguistic analysis based on the accumulated experience in computer linguistics allows simplifying processing of huge amounts of text information and opens up new opportunities for documents processing automating. The problem of finding suitable tools, adapting them to work with texts in the Russian language, and integrating with each other makes difficult to use them both for research and in industrial systems. We present an open source Java framework (TAWT) that provides convenient tools and data structures for the main stages of text analysis which meets modern requirements for performance, reliability, project assembly tools, etc. Examples of automating some technical documentation preparation tasks demonstrate the use of the framework, TAWT can be useful for developers of research tools or applied software for implementing new functions or improving the quality of text processing, as well as for developers of automated tools to reduce routine tasks working with documentation.

1. Introduction

In the overwhelming majority of information systems text is used as the main form of presenting information along with graphics, video, sound, etc. First of all, people exchange messages, read literature, news, reviews and other useful information on the topic they need, write books and articles, prepare reports on work results and documentation, look for information in various databases, references and encyclopedic resources in text form. When solving a variety of tasks applications of automated text analysis can significantly simplify, shorten and speed up manual work.

Achievements in natural language processing (NLP) are increasingly being used to automate text processing in information systems. Developers of computer systems try to bring their functionality closer to the human abilities of understanding a natural text. Various tools are used in a variety of commercial media monitoring software products (Interfax SCAN [1], “Medialogia” [2]), tracking trends in any field, anti-plagiarism systems (Antiplaiat [3], RuCont [4]), and many others. The importance of efforts in this area is also evidenced by the facts of the purchase by Google and Yahoo of the startups Wavii and Summly in 2013, respectively. These companies were involved in the development of automatic summarizing tools. Yandex is also maintaining their works in automatic summarization of web documents based on the search request [5].

All this makes urgent both the creation of new approaches and NLP algorithms and using the existing methods in this area in applied systems to simplify the work with large volumes of text data. In addition, the practical experience in using computer linguistic tools leads to the creation of new tasks and research areas, and finding new applications of existing algorithms and tools.
More and more machine learning methods and neural networks are used today for big data processing, which show good results in solving data recognition and analysis tasks. The application of computer linguistic algorithms for document vectoring requires tools that must be as efficient as possible so that the processing time remains acceptable.

The similar situation is with application of computer linguistic tools in applied industrial systems. On the one hand, linguistic improvements should not slow down the system noticeably; on the other hand, software development tools (libraries, frameworks) supported by modern programming languages, project building tools, etc. must be developed.

This requires specialized development tools that support popular programming languages for industrial software development (Java, .Net, etc.) which are focused on solving applied problems, i.e. are quite high-level from the NLP point of view, implement the stages of linguistic processing, including the requirements for high-load multi-user systems.

Of course, over the years of research, many models had been proposed, a huge number of hypotheses and methods based on them were developed that solved individual problems of text processing. For example, when removing the lexical and morphological ambiguity using the Synan, Trigra and Accopost systems, the accuracy can in some cases reach more than 90% [6], or with certain limitations, for example, the method implemented in the system of Russian-English and English-Russian phraseological translation RETRANS, which allows eliminations the homonyms with 99% accuracy. (The limitation is the basic set of structures, the expansion of which requires manual search and modifications of the system [7-8]. Also, there is a number of semantic models with full or partial automation, for example, the “Meaning-Text” framework [9], the use of concepts for formalizing semantic text processing [10], an approach to semantic search in a collection of documents [11], etc. At the same time, the problem of automatic construction of a formal model of a natural language and using it for solving applied problems has not been resolved. As a result, a fully automatic “understanding” of the text by a computer program is impossible until a number of individual problems of computational linguistics would be solved.

The problem of accurate natural language processing entails the development of various text analysis tools such as Lemmatizer[12], Greeb[13], Stemka[14], pyMyStem3[15], TreeTagger[16], Text Summarization[17], Tools4noobs[18] etc., as well as various toolsets: AOT[19], GATE[20], LingPipe[21], UIMA[22], Texterra[23-24]; in the commercial segment there are Tomita-parser [25], SpeechKit [26] and others by Yandex, the core of Google engine, ABBYY[27] and others. Such variety of solutions is caused by the complexity and lack of a unique solution for the NLP problems. For creating and testing new approaches and hypotheses there is a need of linguistic frameworks and tools to develop new software applications on their basis considering the previous experience in the field of computer linguistics.

Thus, there is the problem of choosing from a wide variety of software tools and libraries that are loosely coupled, operating with different data structures and written in different programming languages. This problem can be solved by a universal solution that is able to provide the minimum necessary text analysis methods and hide the details of the implementation of text analysis algorithms.

The solution should be a framework for the popular cross-platform Java programming language which provides an easy usage for research purposes and for use in developing industrial software systems without the need to study linguistics in depth, the framework should contain implementation of NLP algorithms for the main stages of processing (graphematic, morphological, syntactic-semantic) and provide the ability to start work on the Quick-Start principle, and possibility to replace its components if necessary and the mandatory support of the Russian language.

2. Natural Language Processing Tools Overview

There is no fully automatic “understanding” of the text at the moment. However, there are software solutions that allow performing certain stages of text analysis. Existing tools can be divided into four groups:

1. Linguistic infrastructures with frameworks in their composition (GATE, LingPipe, Apache UIMA, Texterra).
2. Web-resources with sets of text analysis services (META-SHARE, Lapps Grid, Language Grid, CLARIN) [27-29].
3. Tools with application program interfaces (API) (Google, Yandex, ABBYY, RCO, IBM AlchemyAPI [30], Semantria [31], etc.)
4. Libraries and tools implementing one stage or method (AOT, Lemmatizer, Gree, Stemka, pymystem3, TreeTagger).

The GATE system (General Architecture for Text Engineering) is a set of tools for developers and researchers in the field of NLP designed to process any volumes text data [20]. It supports English, French, German, Spanish, Italian, Chinese, Romanian, Hindi, Arabic, etc., but for the Russian language the morphological analysis module and Ontotext repository are available only.

Apache UIMA (Unstructured Information Management applications) is an architecture and a set of libraries and tools for creating, researching and using a wide range of different analysis algorithms, as well as their integration with information retrieval and storage technologies [22]. UIMA supports English, French, German, Italian, Portuguese, Russian, German, Swedish, etc., but the Russian language is supported only by the Apache UIMA AlchemyAPI Annotator keyword extraction module.

LingPipe is a set of developer tools for solving problems of computer linguistics [21]. The solution is focused at industrial applications, i.e. it scales well, has a high processing speed, the results can be easily reused. Arabic, Chinese, Dutch, German, Greek, Hindi, Japanese, Portuguese and Spanish are supported, but there is no support for the Russian language.

Texterra - infrastructure, technology and a set of software tools for NLP with the support for the Russian language [23-24], focused on operating with terms. Texterra can be used as a library of basic NLP algorithms, as an extensible framework, or a cloud service with a huge knowledge base.

The most popular frameworks (GATE, LingPipe, UIMA) are developed for the English language and other foreign languages, while they have little support for the Russian language, which makes it impossible to use them for documents in Russian.

At the same time, attempts to adapt frameworks to the Russian language have no noticeable success: they may require a lot of time, also many methods and approaches are effective for specific language only, providing much worse results for another. For example, the Porter algorithm is effective for the morphological analysis for the English language, but for the Russian it has lower accuracy due to the peculiarities of the languages. There are similar problems with syntactic analysis algorithms because of the differences in the syntax rules of various languages. The recent creation of Texterra framework also confirms the relevance of a framework for the Russian language, but even more oriented on simplifying the applied using.

There are some commercial solutions for the Russian language (Yandex [25], RCO [26], ABBYY [27], Google Search Appliance [32]), however, either their price is high, while often these solutions are monolithic, and it is impossible to get some specific features only, it is also often impossible to add or modify the proposed method. There is the AOT program complex [19], which includes tools for all phases of NLP, however, this complex is being developed for research purposes aiming to improve the quality of NLP, the tools are more suitable for solving a narrow-focused text processing tasks for further analysis of the results obtained, but not for solving applied problems in the industrial systems.

There are software implementations for each NLP stage, however, such solutions are fragmented, that means that it is necessary to create custom interfaces between tools in order to use them together. For example, there are tools for graphematic (Lemmatizer, Greeb, etc.) and morphological (Stemka, pymystem3, TreeTagger, etc.) analysis, but in order to use them together, i.e. the result of one tool is the input to another tool; it is necessary to study all the APIs and develop a common interface for the interaction between them.

The Greeb graphical analysis tool is implemented in the Ruby programming language, “Twitter NLP and Part-of-Speech Tagger” - in Java. Morphological analysis tools: MAnalyzer - in C/C++, as well as AOT complex uses Python/C/C++ and JavaScript [19]. Sharing tools implemented in different programming languages increases the time and cost of their integration, reduces productivity due to
The use of linguistic text analysis tools in the industrial systems usually assumes that they will be introduced into an existing system or into a system under development with already defined technology stack. Currently, the most common programming language according to the research data from various rating agencies is Java [33-34]. Large platforms and text analysis frameworks (GATE, LingPipe, UIMA, Texterra) are also implemented in Java, which also indicates that there is a need of text processing tools in systems implemented in the Java programming language.

In addition, it is quite difficult for software developers to determine the optimal data structures (objects, lists, etc.) for automatic text processing, since this requires wide experience in computer linguistics and understanding of the NLP peculiarities.

3. Framework structure

Basing on the requirements the TAWT (Tools for Automated Work with Text) framework was designed. It consists of a set of software tools [35-37]. The structure and interaction scheme between individual tools are shown on figure 1.

![Figure 1. The structure of the TAWT framework.](image)

All tools use a common connection scheme that is standard for the Java platform: a global binary dependency repository is used with the source code examples and links to artifacts shared on Github [38].

The framework is designed on the theory that a sentence consists of a set of bearing phrases. The bearing phrase is the part of a sentence, the boundaries of which are commas or other punctuation marks and which mandatory contain at least one word with the main(bearing) part of the speech. The main parts of speech are verbs, nouns, participles. Words in the sentence that can have such forms are called bearing words [39].

The Parser tool implements the graphematic stage of NLP. It is based on a set of rules developed with the help of regular expressions. The Parser tool uses different sets of regular expressions to highlight the bearing phrases, sentences, paragraphs in order to correctly recognize the necessary boundaries for the further analysis. The input variable is a string, and the output structure of type List with nested Lists, where the destination node (String) contains the text representation of the recognized word. The List collection allows saving a sequence of words, suggested bearing phrases, sentences, paragraphs, and the nesting level determine the boundaries of words, bearing phrases, sentences, paragraphs. Subsequent analysis assumes that there is no significant difference between complex
sentences and a set of reference turns. The final set of reference turns is formed by the GAMA tool after conducting a morphological analysis.

The JMorfSdk (Java Morphological Sdk) tool implements the morphological stage NLP. JMorfSdk is based on the computer-oriented “Grammar Dictionary of the Russian Language” by A.A. Balzhytsyak modification developed and maintained by the OpenCorpora project [40]. The dictionary contains more than 360 thousand unique words, more than 5 million of word forms, which include rare and new words, as well as the most typical typos to reduce their impact on the analysis.

JMorfSdk is designed for high performance. The complexity of obtaining a set of homofoms of a word and their morphological characteristics is $O(1)$, which is achieved by using hash tables together with bit operations and storing the necessary characteristics as a bit scale. The average speed of the morphological analysis is 900,000 words per second.

A distinctive feature of the tool is the word generation mode. You can get a word according to the given morphological characteristics [35].

**MS tool** (Morphological structures) is a tool for loading, storing, processing and issuing wordforms and word characteristics, contains a description of all data structures. The tool is used in JMorfSdk to convert from the original format in the format used in the framework.

The GAMA tool (Graphematic and Morphological Analysis) aggregates all methods of graphematic and morphological stages NLP, supports the replacement of the tools of graphematic and morphological analysis without recompiling the code.

The RFC (Rules for Compatibility) tool provides a set of implemented word compatibility rules [41]. It includes both simple rules, such as the compatibility of a preposition with a noun in specific case; adjective and noun case consistency, number and gender, etc., as well as more complex rules: verbs, verbal adjectives and adverbs are the control word in relation to a noun. If the noun is in the nominative or accusative case, and there is no preposition in front of it this distance between them should not be more than 4 words, etc. The expanding of this set of rules will be continued.

For example, verb management models’ implementation is planned for the nearest future. This will significantly increase the accuracy of word compatibility processing by considering word collocations, not only at the morphological level, but also at the level of a specific word collocations. For example, the Russian verb “lezhit” (eng. - “lies”) can only control nouns in the nominative and in the instrumental case, and for the word “podshutit” (eng. - “play a trick”) the model will be: “over (preposition) + noun in the dative case”. At the same time, the word “viset” (eng. - “hang”) will have a different control model: “in / on (prepositions) + noun in the prepositional case” or “under / above (preposition) + noun in the positive case” or “near (preposition) + noun in genitive case ”.

The AWF (Ambiguity Words Filter) tool contains a word ambiguity filter based on RFC rules, supports replacing the RFC tool with another model that confirms or denies the compatibility of a pair of input words. According to the results of checks for ambiguity elimination, it was found that it gave the wrong result in no more than 15% of cases [39].

The SP (Syntactic Parser) tool implements the semantic-syntactic stage of analysis, its operation requires a preprocessed text, where each word has certain morphological characteristics. Separate components use the filter to eliminate ambiguity and a set of rules for the words compatibility to establish links between words within the bearing phrase, sentence and paragraph.

Based on the formulated rules, there are algorithms for defining bearing words and establishing links between them, since non-bearing words are mostly dependent from neighboring bearing words, more often on the right, less often on the left of the bearing word. The remaining words are attached to the bearing word to the right [41]. The result of the algorithm is a network or networks of dependencies. The root of which is the predicate for a simple sentence or a subject, if there is no predicate. The ambiguity filter does not always completely remove the ambiguity of all words, for example, when searching for links between words, it is necessary to analyze each combination of homoforms of ambiguous words. Thus, the result of the SP tool may be several dependency networks. In this case, the `getTreeSentence()` method returns all possible variations.
The quality assessment of the word ambiguity elimination is made by comparing the number of ambiguous words for each group of texts before and after applying the filter as well as after establishing connections between words. For texts of various styles, the elimination of ambiguities in test texts (about 100 texts) reaches 50%, and ambiguities are resolved better in fiction texts.

For different text styles, the quality of establishing links between words is different. This is due to the fact that in scientific (about 70% correctly established connections), publicist (about 80%) and technical (about 75%) text there are simpler word structures and related words stand closer to each other, and in fiction texts (result is 50-60%) complex constructions are common, therefore related words can be located at larger distances.

The SPN (Search Possible Notions) tool is designed for searching potential concepts and key phrases based on the analysis of dependency networks from the SP tool. The SPN tool is based on the approach that the most common phrases in natural language texts may be treated as concepts. The concept is one of the main semantic indivisible parts of the sentence.

It is the concept that conveys our mental image (object, phenomenon, event, etc. of the real world) into text form [7]. The concept cannot be divided. For example, the concept of “strategic missile forces” reflects a specific type of troops that stand out among the rest, and if you divide it the concept of smaller units "rocket troops" and "strategic purpose", you get two terms that don’t describe the original type of troops, i.e. they describe other objects.

The speed of semantic-syntactic analysis depends on the style of the text and averages 22.2 sentences per second for publicistic and fiction texts and 66.7 sentences per second for scientific texts. The complexity of the algorithm is \( O(n) \), since the running time of the filter and the search algorithm within the bearing phrase is close to a constant value, and the average speed of the tool when analyzing one sentence does not depend on the text volume. The framework's data structures use internal binary representations of many characteristics, which allows further optimization of memory consumption and reducing time when solving specific problems.

4. Using the framework for applied problems

Computer linguistics could be very useful in document management systems and, in general, in the preparation of different kinds of documentation: project, design, etc. In most cases, the documents are subject to formal requirements on the structure, design, content. Preparation and updating documentation, making packages of documents for software development, any technical solutions, etc. is an extremely time-consuming task.

Technical documentation has a structure defined by one of the standards, for example, for a technical task: GOST 34.602-89, GOST 19.201-78, IEEE STD 830-1998, etc. The possibility of validating the document structure in accordance with the standards is relevant both when training new employees and to simplify the work of experienced specialists, because this allows them to do the initial document validation in an automated mode, and also improves the quality of the technical document.

Such validation has been implemented in the VSS software tool (Validation of Structure by Standard). VSS uses the Parser tool, as well as the Apache POI library for working with .docx files. The graphematic analysis tool allows selecting sections from a chain of characters for subsequent check of the document structure.

The work of VSS was tested on compliance with GOST 34.602-89 on the technical documents from the open database of technical tasks for the software development and/or modernization for the Ministry of Economic Development of the Russian Federation [42].

VSS allows you to check the compliance of a document with a given structure quickly. As a result of VSS work, the most common deviations of the structure of documents from GOST were identified: the wrong level of sections and subsections; changing section titles instead of using subsections.

4.1 Validation of abbreviations in the list of terms and abbreviations

Technical documentation contains a large number of different specific terms and abbreviations that may not be commonly used or have no obvious meaning, depending on the subject. To define a single
meaning for this terms and abbreviations they are placed in a separate section of the document “Terms and Abbreviations”.

However, it may take a long time to create technical documentation, so the developer can forget to specify all the terms in the relevant section or, alternatively, add unnecessary abbreviations that are not used in the text, constant updating of documents is time consuming and often leads to lots of errors. Then there is a need to get a list of abbreviations and abbreviations used in the text automatically, as well as validate them for availability in the section “Terms and Abbreviations”.

Validation using graphematic analysis only without morphological processing yields an inaccurate result, since it is impossible to distinguish between a word written in capital letters from an abbreviation at the graphematic level. Several methods of searching for abbreviations showed that it is necessary to use graphematic analysis, which considers all possible forms of sentences, including all separation symbols, etc. or apply additional post-processing to combine highlighted abbreviations. The criteria that the abbreviation is written in capital letters is mandatory, but not sufficient to determine that the word is an abbreviation. So, without using morphological analysis, the words: “TECHNOLOGY”, “PRINCIPLES”, etc. will be defined as an abbreviation. In order to exclude them from the list, it is necessary to conduct a morphological analysis of the words and to find out that the morphological characteristics of the word do not contain an abbreviation sign.

4.2 Searching for similar documents
Searching for documents that are similar in meaning is relevant if there is a large technical documentation base available and it is necessary to find already developed analogs for the source document, such as technical requirements, that will simplify the development of a new product or software.

To evaluate the similar documents in search results using the TAWT framework, several search methods were implemented:

1. Statistic word search with the Parser tool without using morphological analysis. The peculiarity is that the method is implemented very quickly, however, it has obvious weak point: words in different forms will be considered unique, for example, the words “systems” and “system” will be treated as different words.
2. Statistic word search using morphological processing involves using the GAMA tool. The use of morphological analysis allows bringing each word to the initial form, so the words "system" and "system" will be given to "system" and such words will be considered as one unique word.
3. Statistic search by keywords. The main feature of the keyword search is excluding the other words that do not make sense from a semantic point of view.
4. Statistic search by key phrases. The peculiarity of this search is that the key element is considered to be a word combination that is more precise than a single keyword. This is implemented using the SPN tool, which establishes syntactic links between words and highlights key phrases.

Statistic search counts the number of key elements (words, keywords, key phrases, etc.) in the text and the successive receipt of the intersections of key elements in the source text and other texts on this subject.

Technical documents from software automation, applied software development, design of machine parts, as well as documents that are not relevant to technical documentation, such as reports, descriptions and reviews were used to evaluate the approaches.

Analysis of the results showed that statistic search by words without morphological processing, with morphological processing and by keywords give approximately the same result, however, the application of the search by key phrases eliminates documents that are not suitable for the content, and more precisely separates similar documents from the others, thereby significantly reducing the time required for subsequent manual analysis.
When analyzing the results, it was found out that among the most appropriate documents found there were documents for the previous version of the automated system in the original documents, as well as technical documents for the common development and development of subsystems.

The selected criteria (the number of elements found to the number of elements in the source text) is not optimal, because there is a probability that if the document will be many times larger than the original one, then the number of matching elements may be large, but not because of the similarity, but because of the difference in volume. Therefore, this criterion is suitable for processing approximately equal in volume documents, in other cases it is necessary to use “weights” depending on the features of the analyzed documents.

4.3 Document summarization

In addition to the automatic filtering of inappropriate documents, another way to speed up the search and analysis of the technical documentation is to reduce the volume of the text without significant loss in meaning. Methods of annotation and summarization are used for this purpose. Annotation is a brief description of the text content. Summarization is the construction of the text summary with the disclosure of its main content in all aspects concerned. The summarization provides an answer to the question about what is said in the text, as well as what basic information is contained in it [43].

Thus, summarizing a large document, compressing it and obtaining a brief representation that will reflect the document basic information, allows familiarizing with the main sense of the document and its main ideas, which in turn speeds up the search for the necessary documents.

A library was developed containing a number of different summarizing methods based on the JMorfSdk tool, the most efficient method according to research is the united method [44]. This library was used for development of the “Automatic text summarization service” [45].

To simplify the search for documents which are similar to the given technical specifications for the new information system modules development, abstracts were obtained on automatically selected documents. The technical specification has 90 pages, and contains general information, the objectives of the revision, the characteristics of the automation object, the requirements for the system and for the commissioning of the revisions. There are all of the listed items in brief on the resulting 7-page abstract, which allows a person to determine whether it is suitable for further work quickly, or which parts need to be changed, compare and check the data, etc.

5. Framework further development and software applications based on the framework

The use of linguistic analysis allows increasing the speed and quality text documents processing. For example, it allows identifying abbreviations better, while additional research is needed to identify various ways to shorten words, i.e. identifying not only abbreviations in the form of simple abbreviations, but also abbreviations of a more complex type, for example, word cuts: “pr-n” - “production”, “i.e. - “that is,” etc.

The main areas of the TAWT framework application for automating the processing of technical documents are:

1. Automation of stand-alone applied problems using computer linguistics tools considering the documents in a particular field, company, etc.

2. Development of integrated solutions based on the use of stand-alone tools for the analysis of text documentation.

The use of linguistic analysis not only opens up new opportunities for automating the processing of technical documents, but also defines new research areas and tasks of computer linguistics. Thus, the main directions of development of the TAWT framework are:

1. Improvement of semantic-syntactic analysis tools by adding new rules based on word management models.

2. Development of new tools for the framework, adaptation of keyword extraction tools, named entities recognition, expansion of abbreviations, and text summarizing.
3. Conducting research in the field of computer linguistics to identify reliable search criteria for similar text documents with regard to their features (volume ratios, number of keywords, text style, etc.).

Separate tools of the developed TAWT framework were successfully used in research software for collecting text corpora, selecting named entities, expanding abbreviations, building a semantic network using large dictionaries, keyword highlighting utilities, text summarizing libraries, etc. All the developed tools showed good quality of recognition of named entities (more than 90%), the correct generation of words (about 70%) in decoding abbreviations (including new methods of improvement of abbreviation decoding algorithms), improving the quality of keyword extraction by 8% compared to well-known methods [46-47], implementing the joint summarization algorithm which improves the quality of abstract construction by an average of 15% [45]. All the developed tools and libraries are available on the portal “Automated text analysis” [48].

The linear dependence of the processing time on the number of words allows its use for analyzing large volumes of text data. MS and JMorfSdk tools are used to preprocess texts in an ensemble of classifiers based on machine learning and neural networks [47], to obtain morphological markup of the body of texts [40], for secondary processing of messages from social network in the “FraudHunter” system which detects fraud messages with recognition accuracy about 97% [49].

Using the TAWT framework tools, a number of software applications have been developed. Automatic summarizing service for texts in the Russian language [45]: provides the ability to select the method of constructing a summary, specify the desired volume of it in percent of the volume of the source text, and extract the keywords. Plugin for monitoring and recognizing fraud messages on the social network VKontakte [49]. This Google Chrome browser plugin and its back-end allows you to receive notifications when receiving suspicious messages before they are read and decide on performing requests or actions from the message based on the warning level received. Context synonyms obtaining service [50]: provides the ability to obtain lists of synonyms to the entered word in Russian for each topic chosen by a user. FriendFinder [51] application provides the ability to search for people by interests among the friends in the social networks VKontakte and Facebook, build a chain of friends to a found the desired person. TouristHelper 2.0 application [52]. The Android application allows extracting the text from a QR code or from a web page from QR-link, and simplifying it by reducing the amount of text and highlighting the keywords in order to read and understand the information quickly.

The developed TAWT framework contains the main tools for graphematic, morphological, semantic-syntactic analysis for the Russian language, and data structures for working with the results of each stage of analysis. Open source, ready-made set of data structures and availability in the global Maven repositories make the use of the framework similar to any modern Java-library or framework. Independence from external data sources allows the use of TAWT in software systems of any architecture.

6. Conclusion

The developed framework is a tool for implementing algorithms of NLP at different levels, solving applied tasks and verifying various computer linguistics hypotheses quickly. TAWT is an open source cross-platform solution focused on rapid deployment and industrial use.

All TAWT tools operate with a single ready-made data structures, which excludes the need to develop own structures for storing graphematic, morphological, syntactic analysis results and allows us to store only essential part of intermediate results, in many cases using only binary identifiers.

The main task of the framework is to provide data structures and an extensible set of tools for the three main levels of NLP: graphematic, morphological, semantic-syntactic, united by a common architecture that meets modern software development requirements. The framework can be used not only to accelerate the development of tools for linguistic research, but also in creation of any kind of computer software systems adding the linguistic features and simplifying the work with text information.
The use of linguistic analysis allows improving the quality of text data processing in many areas and opens up new opportunities for automating the text documents processing, using the computer linguistics experience and peculiarities of the tasks. The advantages of applying the graphematic and morphological analysis tools were shown by the example of document structure validation and validation of the list of abbreviations. The search for similar documents and decreasing the size of found documents significantly speeds up their analysis and simplifies the work with large sets of documents.

During the solving the problem of searching for similar documents, it was revealed that in addition to the use of linguistic methods, it is also necessary to conduct further research in the field of computer linguistics in order to identify more reliable criteria for searching similar documents by meaning considering their details. Thus, the wider use of linguistic analysis contributes to the emergence of new tasks and areas of research in computer linguistics itself.

The framework is supported and developed, it was used to create other applications using NLP as one of the features: the text summarization service, the service for searching people by interests in social networks, the service for selecting context synonyms, etc. All services are available at [48].

The TAWT’s distinguishing feature is its ease of deployment into existing industrial systems through the use of Java platform standards: the object model and exception handling, the ability to be used in multi-threaded applications, and availability in global Maven repositories. This allows using the framework in any software system along with any other libraries, without requiring changes in the system architecture, the way how the project is built, deployed and maintained.

The developed cross-platform framework is useful for developers of text analysis tools for scientific research, developers of Java applied software for implementing new functions or improving the quality of text processing by introducing linguistic methods, as well as for developers of automated tools to reduce routine work with any type of documentation.

References
[1] Scan - system of comprehensive news analysis, https://scan-interfax.ru [Accessed:2020-12-12]
[2] Software products of the company Medialogia http://www.mlg.ru/ [Accessed:2020-12-12]
[3] Internet service Antiplagiat, Anti-plagiat company https://www.antiplagiat.ru [Accessed:2020-12-12]
[4] System RuContext, RuCont company http://text.rucont.ru [Accessed:2020-12-12]
[5] P Braslavski and I Kolychev 2005 Automatic Query-Based Summarization of Web documents Internet-Mathematics journal pp. 485–501 (in Russian)
[6] A Sokirko and S Toldova 2005 Comparison of the effectiveness of two methods of removing lexical and morphological ambiguity for the Russian language (The hidden Markov model and the syntactic analyzer of name groups) Internet-Mathematics journal pp. 80–94 (in Russian)
[7] G Belonogov and Y Zelenkov and others 1993 Automation of collecting and maintaining dictionaries for systems of phraseological computer translation of texts from Russian into English and from English to Russian Nauchno-tekhnicheskaja informacija vol.2 pp. 16–21 (in Russian)
[8] G Belonogov and Y Zelenkov and B Kuznetsov and others 1993 An interactive system of Russian-English and English-Russian machine translation of multi-thematic scientific and technical texts Nauchno-tekhnicheskaja informacija vol.2 pp. 20–27 (in Russian)
[9] I Bolshakov and A Gelbukh 2011 The Meaning-Text model: Thirty years after International Forum on Information and Documentation Mexico City, Mexico
[10] A Razorenov and V Fomichev 2017 A New Approach to Formalization of Instructions: Semantic Processing Based on the Theory of K-representations Information Technology vol.1 pp. 3–14
[11] N Chirkova and K Vorontsov 2016 Additive Regularization for Hierarchical Multimodal Topic Modeling A New Approach to Formalization of Instructions: Semantic Processing Based on the Theory of K-representations Journal of machine learning and data analysis pp. 187–200
[12] Official site lemmatizer https://wiki.de.dariah.eu/display/TextGrid/Lemmatizer [Accessed:2020-12-12]
[13] Official site lemmatizer greeb https://github.com/dustalov/greeb [Accessed: 2020-12-12]
[14] Korobov M 2015 Morphological Analyzer and Generator for Russian and Ukrainian Languages International Conference on Analysis of Images, Social Networks and Texts
[15] Official site PyMyStem3 https://pypi.python.org/pypi/pymystem3 [Accessed: 2020-12-12]
[16] Treetagger. Ludwig-Maximilians-Universität München (lmu) http://www.cis.uni-muenchen.de/~schmid/tools/TreecTagger [Accessed: 2020-12-12]
[17] Official site of text summarization http://textsummarization.net/text-summarizer [Accessed:2020-12-12]
[18] Official site of tools4noobs https://www.tools4noobs.com/summarize [Accessed:2020-12-12]
[19] Official site AOT http://aot.ru [Accessed:2020-12-12]
[20] Gate - general architecture for text engineering, https://gate.ac.uk [Accessed:2020-12-12]
[21] Official site of LingPipe system http://alias-i.com [Accessed:2020-12-12]
[22] Official site of Uima, Apache Uima Project http://uima.apache.org [Accessed:2020-12-12]
[23] Texterra: a toolkit for text mining https://texterra.ispras.ru [Accessed:2020-12-12]
[24] D Turdakov and N Astrakhantsev and Y Nedumov and others 2014 Texterra: A Framework for Text Analysis Programming and Computer Software pp. 288–295
[25] Official site Yandex Tomita Parser https://tech.yandex.ru/tomita [Accessed:2020-12-12]
[26] Official site RCO http://www.rco.ru [Accessed:2020-12-12]
[27] ABBYY Compreno https://www.abbyy.com/ru-ru/isearch/compreno [Accessed:2020-12-12]
[28] The language applications grid, us national science foundation office of cyberinfrastructure http://www.lappsgrid.org [Accessed:2020-12-12]
[29] Clarin - European research infrast. for language resources https://www.clarin.eu [Acc.:2020-12-12]
[30] Official site of AlchemyApi service https://www.ibm.com/cloud/blog/announcements/bye-bye-alchemyapi [Accessed:2020-12-12]
[31] Official site of lexalytics intelligence platform https://www.lexalytics.com [Accessed:2020-12-12]
[32] Google Search appliance https://enterprise.google.com/search https://www.lexalytics.com
[33] Tiobe - the software quality company https://www.tiobe.com/tiobe-index [Accessed:2020-12-12]
[34] PyPI popularity of programming language http://pypl.github.io/PYPL.html [Accessed:2020-12-12]
[35] E Politsyna and S Politsyn and A Porechny 2018 Development of the cross-platform library of morphological analysis of the Russian language text for industrial software Proceedings of the CEE-SECR’18 New York, NY: ACM pp 1–10
[36] E Politsyna and S Politsyn and A Porechny 2018 The framework for hypothesis verification and analysis of natural language processing for the Russian language Supplementary Proceedings of the Seventh International Conference on Analysis of Images, Social Networks and Texts AIST-SUP 2018 Aachen, Germany: CEUR WorkShop Proceedings pp 25–33
[37] Porechny A 2018 Implementation of the cross-platform framework for automatic processing of natural language text The XLV Gagarin Science Conference Moscow, Russia pp 112–113 (In Russian)
[38] TAWT framework and tools sources https://github.com/jalexpr/JMorfSdk [Accessed:2020-12-12]
[39] G Belonogov 2008 Theoretical Problems of Computer Science Proceeding of Plekhanov’s Conference vol 2 pp. 53-60 (In Russian)
[40] OpenCorpora dictionary official page http://opencorpora.org [Accessed:2020-12-12]
[41] Porechny A 2017 Implementation of the search for concepts by selecting word combinations from text) Proceedings of the Conference Problems of computer linguistics and typology Voronezh, Russia: VSU pp 108–111
[42] Official server of state authorities of the Russian federation http://aisup.economy.gov.ru/pubportal [Accessed:2020-12-12]
[43] L. Markushevskaja and Ju Capaeva 2008 Annotation and abstracting: guidelines for students
St. Petersburg, Russia: ITMO

[44] Kasatkina A 2019 Development of the joint method of automatic text summarization Computer
Science: problems, methodology, technologies (Moscow, Russia: MAI) pp 355–356 (In Russian)

[45] Automatic text summarization service http://abstracts.textanalysis.ru [Accessed:2020-12-12]

[46] Ivashchenko M 2018 Analysis of automated keyword extraction methods in natural language texts
Computer Science: problems, methodology, technologies vol 6 Voronezh, Russia: VSU pp 355–356 (In Russian)

[47] Samokhin A 2019 Analysis of algorithms and the extending the tools of abbreviations removal
for the russian language Computer Science: problems, methodology, technologies vol 1
Voronezh, Russia: VSU pp 355–356 (In Russian)

[48] Portal “Automated text analysis” http://textanalysis.ru [Accessed:2020-12-12]

[49] Y. Petrash and D. Tikhonova 2019 Development of the tool for fraud messages identification in
the social network “VKontakte” Computer Science: problems, methodology, technologies
vol 1 (Voronezh, Russia: VSU) pp 355–356 (In Russian)

[50] Official site of Thematic synonyms service http://synonyms.textanalysis.ru [Accessed:2020-12-12]

[51] Official site of FriendFinder application http://friendsfinder.textanalysis.ru [Accessed:2020-12-12]

[52] Official site of Tourist Helper 2.0 https://play.google.com/store/apps/details?id=ru.textanalysis.touristhelper [Accessed:2020-12-12]