Bulletin of TUIT: Management and Communication Technologies

Volume 4 *2021-uchinchi son* Article 1

April 2021

Linguistic features tokenization of text corpora of the Uzbek language

I.I. Bakaev
"Bulletin of TUIT: Management and Communication Technologies", bakayev2101@gmail.com

Linguistic features tokenization of text corpora of the Uzbek language

PhD student. Bakaev I.I.

Tashkent University of Information Technologies, 108, Amir Temur ave., Tashkent, 100200, Uzbekistan

bakayev2101@gmail.com

Abstract. *The article deals with the topic of tokenization of text corpora of the Uzbek language, which should take into account the linguistic features of the spelling of the language. The types of words are analyzed and a mathematical model of word formation for complex, paired, repeated, and compound words is proposed. And also based on the models developed a finite state machine to demonstrate the spelling rules of the language, and the rules are implemented by regular expressions.*

Keywords. *Token, tokenization, regular expressions, finite state machine, Kleene star, Kleene plus, corpora.*

1. Introduction. Tokenization is one of the most important stages of processing text corpora, which determines the input sequence of characters into recognized groups of so-called tokens or tokens. To date, there are different methods of tokenization for inflectional and agglutinative languages of the world. For example, rule-based or feature-based methods, lexical methods, static methods, and odd methods. Although some of them are implemented in high-level programming languages, when splitting the text into words, they allow semantic errors. These errors are related to words of complex structure that are written separately but are considered to be a single word. Most often, complex words are found in texts related to agglutinative languages. Such languages include such as Kazakh [1], Kyrgyz[2], Uyghur[3], Tatar[4], Bashkir[5], and including Uzbek [6]. In these languages, when tokenizing, the incorrect definition of complex words in texts leads to a problem of morphological analysis. As a result, problems related to morphological analysis appear errors. These include tasks such as machine translation, searching and extracting information from text, spell checking and syntactic definition of sentences, etc.

In the world of natural language processing, the process of splitting text into words is interpreted as text segmentation and text tokenization, but although they mean the same significance. There are many works that are related to the problems of splitting the text into words, below we describe some of them.

2. Related works. In the article [7], a tokenization algorithm for extracting information from a large text is proposed. The algorithm is based on checking for spaces or a newline character, then the word is assigned to an array of tokens and spaces are removed (or assigned as a character), this before the conditional loop continues until it reaches the end of the line.

The paper [8] describes the general problems of text tokenization. The author claims that the patterns of tokenization of the text, which is determined by a regular relation or a finite converter, are able to characterize the complexity and ambiguity of punctuation conventions in all languages of the world. The proposed algorithm is specialized for special properties of text streams: they are usually quite long, but can be represented by finite automata with a single (acyclic) path. Template tokenization of text with finite converters can be described using regular expressions, which explicitly specifies the boundaries of the tokens.

Researchers [9] applied a new strategy of stochastic tokenization, which segments the supply of tokens stochastically and trains the classification model with different

segmentation. In training, the model first segments sentences into tokens with the language model and then passes the tokenized sentences to the neural text classifier.

The article [10] describes comparative analysis and dictionary-based matching algorithms and machine learning for the text segmentation task. The dictionary-based matching algorithm scans the input characters from the strings and checks for matching with the dictionary set. The algorithm relies on checking the longest and maximum string matching. A feature table is generated for machine learning algorithms analyzing Thai language texts. The experiment shows that the dictionary-based algorithm and the support vector machine algorithm (with an accuracy of 95.79% as measured by F1) gave the best result when segmenting Thai texts.

The researcher [11] studied the characteristics of the Chinese language and selected a conditional random field model, which is trained using quasi-Newton algorithms to perform sequence labeling. Additionally, a set of attributes for the Chinese language has been created. The algorithm is applied in the academic buildings of CityU (City University of Hong Kong) and CTB (Pennsylvania Chinese Library of Trees) which contains 36228 and 23444 Chinese sentences. For these cases, the algorithm for measure F gave 92.68% and 94.05%, respectively, for CityU and CTB.

In [12], a tokenization system was developed for the Arabic language, which is based on the spelling rules of the language. When analyzing Arabic texts, the main syntactic units are identified, such as verbose expressions or punctuation marks, spaces, and words that consist of bases with or without colitis. A system based on the technology of finite states has been developed. The interaction of the tokenizer with other converters and how verbose expressions are identified and separated is demonstrated. The filtering of incorrect tokenization texts and how they are marked is shown.

In the scientific work [13], a tokenization approach is proposed for the dialects of the Kurdish language Sorani and Kurmanji. These dialects use different alphabets such as Latin and Arabic. Therefore, for the tokenization of texts of different alphabets, a lexicon and a morphological analyzer are proposed. Word tokenization tasks consist of several steps: Text preprocessing, compound word tokenization, word tokenization, and morphological analysis. The results show that the accuracy of the base level for composite forms, i.e., is 100%, respectively.

In [14], a LexToPlus tokenizer for the Thai language is proposed, which is based on the dictionary to detect existing terms in the dictionary. When tokenizing, text corpora from Twitter are used. In the analysis, the tokenizer gave a result with an accuracy of 96.3% on a set of text data. The effectiveness of the dictionary approach depends on the quality and size of the set of words in the dictionary.

The authors [15] analyzed the possibilities of the NLTK library in Python, which is the tokenization of words. The analysis uses all NLTK tokenizations such as Nlpdotnet Tokenizer, Mila Tokenizer, NLTK Word Tokenize, TextBlob Word Tokenize, MBSP, Word Tokenize, Pattern Word Tokenize, and Word Tokenization. The results of the study proved that the performance of Nlpdotnet Tokenizer is better than other tools.

Researchers [16] described the Sudachi tokenizer for Japanese text, especially for industrial applications. The tokenizer works on the basis of a dictionary that includes 2.6 million words and 1.4 million of them are exactly normalized forms. The researchers conducted an analysis with other developments such as

IPADIC, UniDic, NEologd. The results show that Sudachi is more effective at tokenizing Japanese texts.

The above solutions mean that each natural language requires a separate approach for text tokenization. Therefore, in this article we describe the approach of tokenization of texts for the Uzbek language.

3. Statement of the problem. When it comes to tokenizing texts, you first need to analyze the verbal structure of the language.

In the Uzbek language, the structure of words consists of five types (simple, complex, paired, repeating, compound). When tokenizing texts, the definition of complex, paired, repeating, compound words that will be written in written speech is merged, semi-merged (separated by a hyphen) and separately tab.1 requires special hard work.

Table 1. Types of words by structure

Part of speech	Types of words			
	Complex	Paired	Repeated words	Abbreviations
Noun	Uchburchak O'rta Osiyo	ota-ona	yor-yor	BMT
Adjectives	Sohibjamol	Sog'-salomat	katta-katta	-
Numerals	O'n besh	Bir-ikki	Yuzta-yuzta	-
Pronouns	Mana bu	u-bu	nima-nima	-
Adverb	Shu yerda	asta-sekin	tez-tez	-
Verb	Sotib olmoq javob bermoq	Qo'llab-quvvatlamoq	pishgan-pishgan	-
In written speech	Fused separately	Half fused	Half fused	Abbreviated

To describe the mathematical model of the formation of complex, paired, repeating, compound words, we define the formal elements of the language.

Σ – a set of characters, which consists of a set of Latin letters and an apostrophe sign. $\Sigma = \{A, B, ..., Z, a, b, ..., z\}$;

D – These are sets of numbers. $D = \{1, 2, 3, 4, 5, 6, 7, 8, 9, ...\}$;

ε – An empty string of length 0;

c_{sp} – white space character;

ρ – The spelling mark (- hyphen);

Now we define operations on the language.

The "Kleene Star" operation.

Σ^* – is the set of all strings of finite length generated by the elements of the set Σ with the addition of an empty string.

D^* – is the set of all numbers of finite length generated by the elements

of the set from D.

$D^* = \{D^i : i = 0,1,2,...,n\}$

Operation "Kleene Plus"

$\Sigma^+ = \{A, B, ... Z, a, b, ... z, AA, AB, ...\}$

;

$D^+ = \{D^i : i = 1,2,...,n\};$

W– Word sets, $W = \{w_1, w_2, ..., w_n\};$

Next, we formally describe the mathematical model of education for complex words that are written separately:

+ - concatenation of character strings.

$$w_1 = w_1' + c_{sp} + w_1'', \quad (1)$$

$$w_2 = D^* + \rho + w_2', \quad (2)$$

For example, - hurmat qilmoq (complex verb), - hurmat, - qilmoq, - 2021-yil (complex numeral names), - 2021, -yil.

The following mathematical model is used for paired and repeated words that are written semi-together (separated by hyphens):

$$w_3 = w_3' + \rho + w_3'', \quad (3)$$

$$w_4 = w_4' + \rho + w_4', \quad (4)$$

For example, - ota-ona (paired noun), - ota, -ona, - katta-katta(repeating the adjective), - katta.

In addition, the last mathematical model is used for complex abbreviated words that are written by shortening by the initial letters or by combining the initial element of the first word and the first letters of the following words. For example, such as BMT (Birlashgan Millatlar Tashkiloti) - UN (United Nations), O'ZFA(O'zbekiston Fanlar akademiyasi) - Academy of Sciences of Uzbekistan:

$$w_5 = \Sigma^*, \quad (5)$$

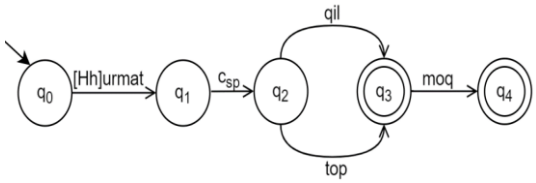
w_5– example, - BMT (complex abbreviated noun).

We need to create an algorithm for text tokenization, which takes into account the types of words according to the structure of the specified models (1)-(5).

4. Solution methods. To divide the text into words, we use a finite automaton (FSM), which can be written as regular expressions(RE). Below is an example for each type of word.

The description of complex verbs of the type w_1 (for example, for the words "hurmat qilmoq" and "hurmat topmoq") can be described using FSM (Figure 1):

FSM in the form of RE ([Hh]urmat)c_{sp}(qil+top)moq

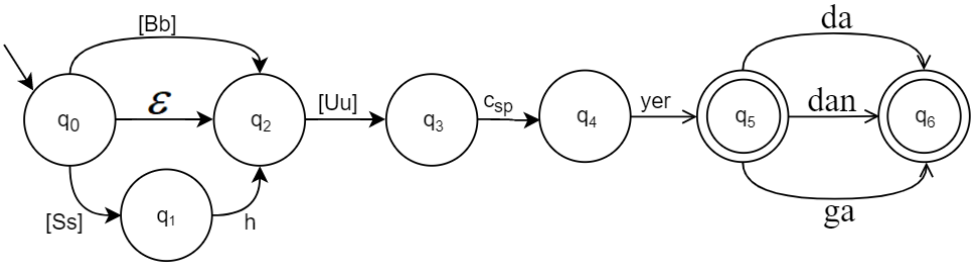


1-Figure FSM for complex verbs of the type w_1.

For complex adverbs of the type w_1 (for example, u yerga, bu yerda, shu

yerdan) can be described using FSM (Fig. 2):

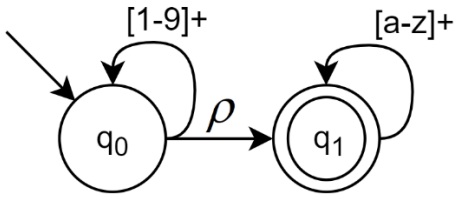
FSM in the form of RE $((s \cdot h) + b)^* \cdot u) \cdot c_{sp} \cdot \Sigma^*$;



2-Figure FSM for complex adverb type w_1.

The description of complex numeral names of the type w_2 (for example, 2021-yil) using FSM has the following form (Fig. 3)

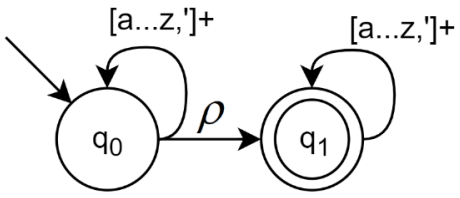
FSM in the form of RE $D^+ \cdot \rho \cdot \Sigma^+$



3-Figure FSM for complex numeral type names w_2.

Generalized description a paired noun type repeating an adjective type name w_4 (for example, ota-ona, katta-katta) can be described using FSM in the following form (Figure 4)

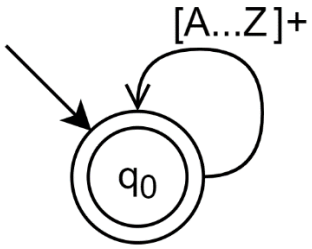
FSM in the form RE $\Sigma^+ \cdot \rho \cdot \Sigma^+$



4-Figure FSM for words of type w_3 and w_4.

The description of complex abbreviated words of the type w_5 (for example, BMT) with the help of FSM has the following form (Fig. 5.)

FSM in the form RE Σ^+



5-Figure FSM for words of the type w_5

Next, we describe the text tokenization algorithm, which consists of two parts.

1) The definition of words of simple, complex, paired, repeating, compound types, which is written in written speech together (w_5), semi-together (w_2, w_3 and w_4). As well as existing templates such as number definition, email address, URL address, and others.

2) The definition of words of complex types, which is recorded separately in written speech (w_1).

Generically, we describe the algorithm for tokenizing the verbal form consists of the following steps:

Step 1. Load the text data (a text file and a string from the clipboard) to the textstr variable of the string type, and assign it to the index=0 variable.

Step 2. Next, cut the i-th character of textstr to a space and add the cut character lines to the templist variable of the list type.

Step 3. Check each templist element with regular expression templates (w_2, w_3, w_4, w_5) and add the appropriate elements to the textlist variable of the list type.

Step 4. Next, we check the regular expression templates (w_1) of the elements i-th and (i+1) th merged from the textlist, if the elements i-th and (i+1) th satisfies the condition, we add the i-th and (i+1) th elements merged to the resultlist. Assign the (i+1) th element to the variable index=i+1 so that the next iteration starts after the (i+1) index. If the i-th and (i+1) th elements do not satisfy the condition, then we add the i-th element to the resultlist.

Step 5. Display the generated word sheet on the screen.

5. Discussion of the results. To test the algorithm, we took text corpora for the genre of fairy tales, legislative documents, and religious works.

№	Chosen corpora	Number of words	Typesof words		
			W_1 и W_2	W_3 и W_4	W_5
1	Fairy-tales	102528	1548	3742	0
2	Law documents	101702	620	1362	1769
3	Religious literary	100946	1534	2624	0

6. Conclusion. Based on this, we conclude that when tokenizing text corpora of the Uzbek language, it is necessary to take into account the linguistic features of the spelling of the language. A mathematical model of word formation for complex, paired, repeated, compound words is proposed. Thus, in this model-based paper, a finite state machine is proposed to demonstrate the spelling rules of the language, and regular expressions are used to implement the rules. The algorithm for tokenizing text corpora is also described.

References

[1] Ó. Aıtbaıuly, *Osnovi kazaxskoy terminologii*. Almata: Абзал-Ай, 2014.

[2] K. A. Biyaliev, *Spravochnik po grammatike kirgizskogo yazika*. Bishkek: Slavyanskiy universitet, 2013.

[3] Z. Rehman, W. Anwar, U. I. Bajwa, W. Xuan, and Z. Chaoying, “Morpheme Matching Based Text Tokenization for a Scarce Resourced Language,” *PLoS One*, vol. 8, no. 8, 2013, doi: 10.1371/journal.pone.0068178.

[4] A. R. Raximova, “Strukturnie i semanticheskie osobennosti slojnix slov, xarakterizuyushix cheloveka, v tatarskom yazike,” *Uchenie Zap. Kazan. Univ.*, vol. 157, no. 5, pp. 205–217, 2015.

[5] L. M. Xusainova, “Pravopisanie slojnix slov-kalek v bashkirskom yazike,” *Vestn. ChGPU im. I. Ya. Yakovleva*, vol. 1, no. 97, pp. 60–65, 2018.

[6] M. B, X. O’, and A. N, *O’zbek tilidan universal qo’llanma*. Toshkent: Akademiknashr, 2019.

[7] J. Joseph and J. R. Jeba, “Information Extraction using Tokenization and

Clustering Methods," *Int. J. Recent Technol. Eng.*, vol. 8, no. 4, pp. 3690–3692, 2019, doi: 10.35940/ijrte.d7943.118419.

[8] K. Ronald M, "A Method for Tokenizing Text," *Complex. Educ. Inq. into Words, Constraints Context.*, no. January 2005, pp. 55–64, 2005.

[9] T. Hiraoka, H. Shindo, and Y. Matsumoto, "Stochastic tokenization with a language model for neural text classification," in *ACL 2019 - 57th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference*, 2020, no. January 2019, pp. 1620–1629, doi: 10.18653/v1/p19-1158.

[10] C. Haruechaiyasak, S. Kongyoung, and M. Dailey, "A comparative study on Thai word segmentation approaches," *2008 5th Int. Conf. Electr. Eng. Comput. Telecommun. Inf. Technol.*, vol. 1, pp. 125–128, 2008.

[11] A. L. F. Han, D. F. Wong, L. S. Chao, L. He, L. Zhu, and S. Li, "A study of Chinese word segmentation based on the characteristics of Chinese," in *International Conference of the German Society for Computational Linguistics and Language Technology, (GSCL 2013)*, 2013, vol. 8105 LNAI, no. May, pp. 111–118, doi: 10.1007/978-3-642-40722-2_12.

[12] M. A. Attia, "Arabic tokenization system," in *SEMITIC@ACL 2007*, 2007, no. June, p. 65, doi: 10.3115/1654576.1654588.

[13] S. Ahmadi, "A Tokenization System for the Kurdish Language," in *VARDIAL*, 2020, pp. 114–127.

[14] C. Haruechaiyasak and A. Kongthon, "LexToPlus: A Thai Lexeme Tokenization and Normalization Tool," in *Proceedings of the 4th Workshop on South and Southeast Asian Natural Language Processing*, 2013, pp. 14–18.

[15] V. S and J. R, "Text Mining: open Source Tokenization Tools – An Analysis," *Adv. Comput. Intell. An Int. J.*, vol. 3, no. 1, pp. 37–47, 2016, doi: 10.5121/acii.2016.3104.

[16] K. Takaoka, S. Hisamoto, N. Kawahara, M. Sakamoto, Y. Uchida, and Y. Matsumoto, "Sudachi: A Japanese tokenizer for business," *Lr. 2018 - 11th Int. Conf. Lang. Resour. Eval.*, pp. 2246–2249, 2019.