

# Computer linguistic system architecture for Ukrainian language content processing based on machine learning

Victoria Vysotska

Lviv Polytechnic National University, Stepan Bandera 12, 79013 Lviv, Ukraine

## Abstract

The general architecture of computer linguistic systems (CLS) is developed based on the main processes of processing information resources such as integration, maintenance and content management, as well as using methods of intellectual and linguistic analysis of text flow using machine learning technology. The IT of intellectual analysis of the text flow based on the processing of information resources has been improved, which made it possible to adapt the generally typical structure of content integration, management and support modules to solve various of natural language processing (NLP) problems and increase the efficiency of CLS functioning by 6-9%. The main NLP methods based on regular expression (RE) matching with patterns in grapheme and morphological analyses of Ukrainian-language texts are described. NLP methods based on pattern-matching regular expressions have been improved, which made it possible to adapt methods of text tokenization and normalization by cascades of simple substitutions of regular expressions and finite state machines. The main valid operations of regular expressions are defined as union and disjunction of symbols/strings/expressions, number and precedence operators, as well as anchors as special symbols for identifying the presence/absence of symbols in RE. The main stages of tokenization and normalization of the Ukrainian text by cascades of simple substitutions of regular expressions and finite state machines are defined. The morphological analysis (MA) method of the Ukrainian-language text based on word segmentation and normalization, sentence segmentation and modified Porter's stemming algorithm was improved as an effective means of identifying lem affixes for the possibility of marking the analyzed word, which made it possible to increase the accuracy of keyword searches by 9%. Algorithms for word segmentation and normalization, sentence segmentation, and Porter's modified stemming are implemented and described as an effective way of identifying lem affixes for the possibility of marking the analyzed word. Unlike the classic Porter algorithm (it does not have high accuracy even for English-language texts), the modified one is adapted specifically for the Ukrainian language and gives an accurate result in 85-93% of cases, depending on the quality, style, genre of the text and, accordingly, the content of CLS dictionaries. The algorithm for the minimum editorial distance of lines of Ukrainian texts is described as the minimum number of operations necessary to transform one into another.

## Keywords

natural language processing, Ukrainian text, NLP, computer linguistics, machine learning

---

MoDaST-2024: 6th International Workshop on Modern Data Science Technologies, May, 31 - June, 1, 2024, Lviv-Shatsk, Ukraine

 victoria.a.vysotska@lpnu.ua (V. Vysotska)

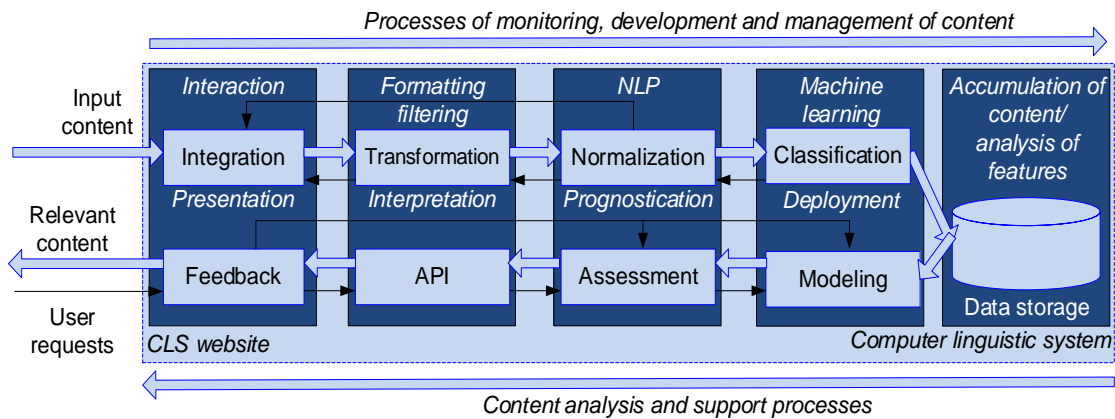
 0000-0001-6417-3689 (V. Vysotska)



© 2024 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

## 1. Introduction

Let's consider the architectural patterns of CLS design based on supporting the life cycle of the ML model for monitoring/managing the pipeline (information flow) of content (Fig. 1) [1]. The standard content processing pipeline implements an iterative process consisting of the stages of creating and deploying the machine learning (ML) process [2-5]. The process of monitoring/managing the content pipeline should still consist of additional stages to improve the quality/efficiency/efficiency of NLP problem solving [6-9]. At the construction stage, raw integrated content is filtered from noise/duplicates and formatted into a suitable form for further processing/management, conducting experiments on it, transferring it to ML models for classification/clustering/prediction/evaluation, etc. [10-12]. At the stage of content analysis and support, the content is deployed to determine the best ML model for making assessments/forecasts that directly affect the regular user and target audience.

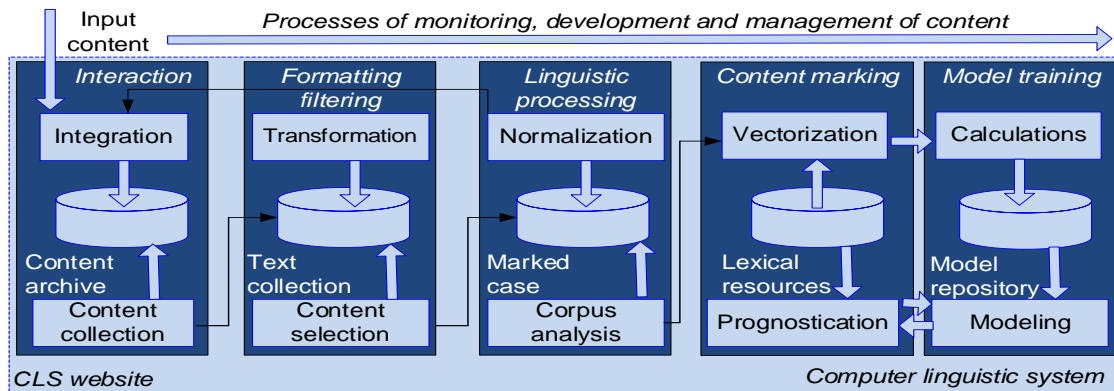


**Figure 1:** CLS content pipeline monitoring/management scheme

## 2. Related works

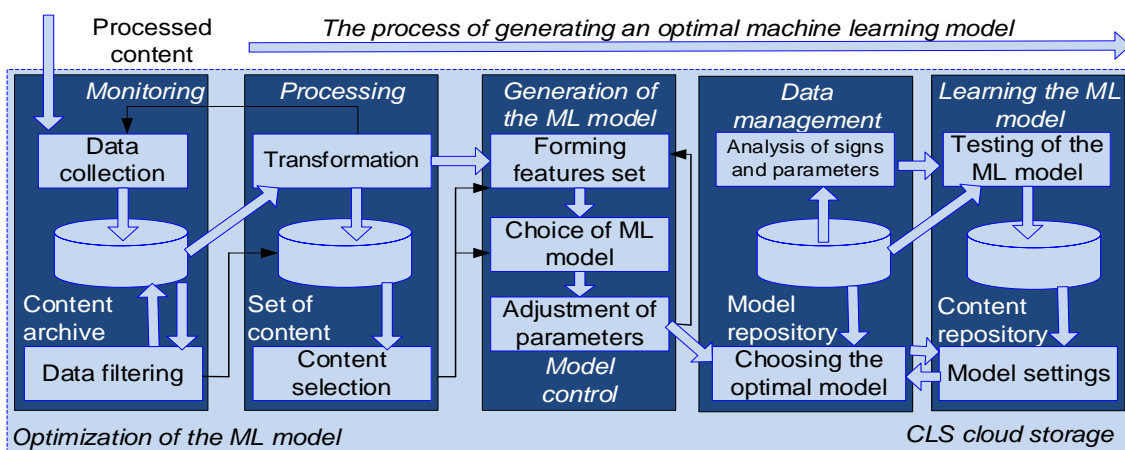
Based on Feedback and model output, the target audience interacts with CLS, which facilitates the adaptation of the selected learning model. Five stages of related processes define the basic architectural principles for building a typical CLS. The processes of monitoring, processing and managing content are interaction, formatting/filtering, NLP, machine learning [13-15] and data accumulation in DS. For content analysis and support processes, respectively, these are feature analysis, deployment, prediction, interpretation, and content/result presentation. At the interaction stage, a set of rules for integrating content from multiple reliable sources at certain time intervals is necessary. Also, in parallel, a set of rules for checking the data entered by the CLS user is required as a preliminary stage for the formatting/filtering stage according to a collection of rules pre-set by the moderator and content from DS [16-21]. The next stage of NLP is a preparatory intermediate stage for machine learning and data accumulation. The machine learning stage can take various forms from SQL queries to various software modules. The support process is easier to implement than the management stage, provided that the latter is implemented correctly, especially during NLP analysis, in which additional lexical resources and artefacts

(dictionaries, translators, regular expressions, etc.) are created, on which the effectiveness of CLS functioning directly depends (Fig. 2) [1-3].



**Figure 2:** Scheme of processing the CLS content pipeline

The process of transition from raw text to a developed machine-learning model consists of a sequence of additional content transformations. First, the input textual content is transformed into an input corpus as a collection of texts, accumulated and stored in the DS. The incoming content is further grouped, filtered, formatted, linguistically processed, marked, normalized and converted into vectors for further processing. In the final transformation, the model/models (Fig. 3) are trained on the vector corpus, and a generalized presentation of the original content is created for further use in solving a specific NLP problem [1-6]. An ML-based CLS architecture with accelerated or even automatic model generation should support and optimize content transformation with ease of testing and tuning. The process of generating an optimal ML model is a complex cyclic algorithm, the main stages of which are the formation of a collection of features, model selection, and hyperparameter adjustment. After each iteration, the results are evaluated to determine the optimal collection of features, models, and parameters for solving a specific NLP problem with the appropriate input data [1-6].



**Figure 3:** The process of forming and optimizing a machine-learning model

According to [1-6], there are 3 main areas of statistical ML: a class of models, a form of a model, and a trained model. The class of models defines the relationship between the variables and the formed goal (for example, a linear model, a recurrent neural network, etc.). A model form is a specific component of a model: a collection of features, an algorithm, or a collection of hyperparameters. A trained model is a form of model that is trained on a specific data set and adapted to make predictions. CLSs consist of many trained models built during their selection, which creates and evaluates model shapes.

### 3. Materials and Methods

Any natural language text is initially a collection of non-random unstructured data as input content to CLS. But usually, the text is formed based on certain linguistic rules for the possibility of understanding these data. The purpose of the integration module is to transform this collection of non-random unstructured data into structured/semi-structured fields (records) or markup for convenient interpretation by CLS modules. ML methods (for example, learning by a teacher) allow you to train (and retrain) statistical models as the language changes during NLP processes. By generating ML models on context-sensitive corpora, CLSs can apply narrow semantic values to improve accuracy without the need for additional interpretation.

Formally, the ML model of the Ukrainian language has to supplement the input incomplete phrase with missing words/phrases that are most likely to complete the content of the statement according to the previous text (context analysis for further guessing/predicting the meaning). Usually, a competently and correctly constructed text is predictable based on its coherence. Calculation of the entropy (degree of uncertainty/unpredictability) of the probability distribution of the model of the Ukrainian language measures the degree of predictability of the text. Thus, unfinished phrases *Київ - столиця...* [Kyiv - stolytsya...] (Kyiv - the capital...) and *сонце сходить на...* [sontse skhodyt' na...] (the sun rises on...) have low entropy and statistical speech models are highly likely to guess the continuation of *України* [Ukrayiny] (Ukraine) and *сході* [skhodi] (the east), respectively. And expressions with high entropy like *ми йдемо в гості до...* [my ydemo v hosti do...] (we go to visit...) and *я зустрів сьогодні...* [ya zustriv s'ohodni...] (I met today...) offer many continuation options (parents, friends, neighbours, colleagues are equally likely without analyzing the previous context). Speech models can make inferences or identify connections between lexemes. Formally, the model uses context to identify a narrow decision space from a set of a small number of options. The application of statistical ML methods (with and without a teacher) allows the generation of speech models for extracting meaning from texts to support its predictability. First, the characteristic features of the content are identified to predict the goal. Textual data provides many opportunities to extract surface features based on parsing and breaking up sentences/utterances/phrases (e.g. bag of words), as well as based on extracted morphological/syntactic/semantic features. Special attention is paid to linguistic/ contextual/ structural features.

1. An example of the analysis of a linguistic feature can be the identification of the predominant gender in a fragment of the news text (the role of gender) in different contexts [1] to identify gender biases regarding the subject of publications. In the gender analysis of

the text, words in the feminine and masculine gender are used to form a frequency assessment of gender characteristics, i.e.

$$Sing_{GS} = \langle X_{Sentence}, W_{Male}, W_{Female}, W_{Unknown}, W_{Both}, f_{gendetize} \rangle, \quad (1)$$

where  $X_{Sentence}$  is analyzed sentence/expression;  $W_{Male}$  is a set of words with the sign of a man;  $W_{Female}$  is a set of words with the attribute woman;  $W_{Unknown}$  is a set of words with an unknown gender sign;  $W_{Both}$  is a set of words with the sign of a man and a woman;  $f_{gendetize}$  is the operator for identifying the gender class of a sentence.

$$Sing_{GS} = f_{gendetize}(X_{Sentence}, W_{Male}, W_{Female}, W_{Unknown}, W_{Both}), \quad (2)$$

$$Sing_{GS} = \begin{cases} N_{Male} > 0, N_{Female} = 0 \rightarrow male \\ N_{Male} = 0, N_{Female} > 0 \rightarrow female \\ N_{Male} > 0, N_{Female} > 0 \rightarrow both \\ unknown \end{cases}$$

where  $N_{Male}$  is the number of words with the sign of a man in the analyzed sentence  $X_{Sentence}$ ;  $N_{Female}$  is the number of words with the sign female in the analyzed sentence  $X_{Sentence}$ .

It is also necessary to determine the frequency of words, gender signs and sentences in the entire publication:

$$Sing_{TS} = \langle X_{Text}, S_{NG}, N_{Sentence}, W_{NG}, f_{countgender} \rangle, \quad (3)$$

$$Sing_{TS} = f_{countgender}(N_{Sentence}, S_{NG}, W_{NG}, f_{gendetize}(X_{Text}, X_{Sentence})),$$

where  $X_{Text}$  is analyzed publication text;  $S_{NG}$  is a set of numbers of  $X_{Sentence}$  sentences of the analyzed text  $X_{Text}$  marked by gender;  $N_{Sentence}$  is the number of sentences in the analyzed text  $X_{Text}$ ;  $W_{NG}$  is the set of the number of words of each gender characteristic for each marked sentence  $X_{Sentence}$ ;  $f_{countgender}$  is an operator of identification and classification/marketing of all sentences of the analyzed text  $X_{Text}$  by gender based on  $f_{gendetize}$ .

$$Sing_{TS} = \begin{matrix} N_{Sentence} \\ \vdots \\ 1 \end{matrix} \left[ \begin{matrix} S_{NG}[Sing_{GS}] += 1 \\ W_{NG}[Sing_{GS}] += len(X_{Sentence}) \end{matrix} \right] \quad (4)$$

For gender identification, it is necessary to parse the original text of publications with the subsequent marking of sentences and words based on the NLTK library:

$$Sing_{TP} = \langle X_{Text}, S_{Sentence}, N_{Sentence}, W_{Word}, N_{word}, f_{parsegender}, f_{pcent} \rangle, \quad (5)$$

$$Sing_{TP} = f_{pcent}(f_{parsegender}(N_{Sentence}, W_{Word}, N_{word}, f_{countgender}(S_{Sentence}))),$$

$$Sing_{TS} = \begin{matrix} N_{Gender} \\ \vdots \\ k=1 \end{matrix} \left[ \begin{matrix} pcent_k = \left( \frac{W_{NG_k}}{total} \right) * 100 \\ N_{Sentence_k} = S_{NG}[Sing_{GS_k}] \\ print(pcent_k, Sing_{GS_k}, N_{Sentence_k}) \end{matrix} \right]$$

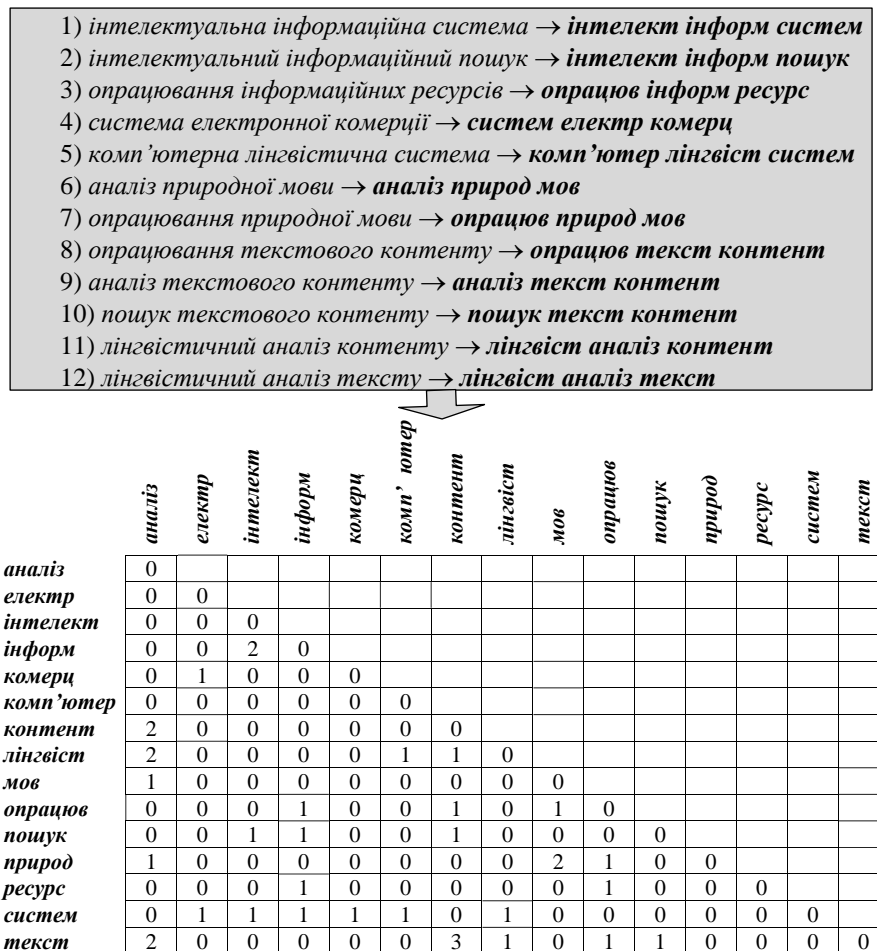
$$total = \sum_{i=1}^{N_{Gender}} W_{NG_i}, \quad S_{Sentence} = \bigcup_i^{N_S} X_{Sentence_i},$$

$$X_{Sentence_i} = \bigcup_i^{N_{WS}} W_{Sentence_i}, \quad W_{Word} = \bigcup_i^{total} W_{Word_i}$$

where  $N_{word}$  is the number of words in the analysed text  $X_{Text}$ ;  $N_{Gender}$  is the number of classifications by gender (in this particular case – 4);  $W_{NG_k}$  is the number of words in sentences of a certain gender sign;  $S_{NG}$  is the set of the number of sentences in the analyzed text of a certain gender sign;  $pcent_k$  is the percentage of publication text belonging to a certain gender sign;  $Sing_{GS_k}$  is a specific gender sign;  $N_{Sentence_k}$  is the number of sentences in the analyzed text of a specific gender sign;  $S_{Sentence}$  is a set of sentences identified by parsing in the analysed text  $X_{Text}$ ;  $W_{Word}$  is a collection of sets of words identified by parsing in each sentence of the analyzed text  $X_{Text}$ ;  $W_{Word}$  is the set of all words of the text  $X_{Text}$ ;  $total$  is the number of all words in the analysed text  $X_{Text}$ . Such a deterministic mechanism demonstrates how the content/frequency of use of words/phrases (especially stereotypical ones) affects the predictability of the content according to the previous context (the gender sign is built directly into the Ukrainian language – every noun has a gender). But speech signs are not always decisive, for example, plural and time are used to analyse language/processes/actions/events in time.

2. An example of the analysis of a contextual feature can be the analysis of moods or sentiment analysis of a text (emotional colouring when discussing a specific topic by a relevant group of people). Usually used in complex analysis of feedback from users, for example, e-commerce, the polarity of messages or reactions to events/phenomena, in social networks or in political/economic discussions/forums, etc. In superficial sentiment analysis, the mechanism of gender classification (positive/negative/neutral coloured word) is usually used. For example, for positive – *чудовий* [chudovyyu] (wonderful), *прекрасний* [chudovyyu] (beautiful), *правдивий* [pravdyvyyu,] (true), negative – *лінивий* [linyvyuu] (lazy), *поганий* [pohanyu] (bad), *дратівливий* [drativlyvyuu] (annoying), and neutral – *білий* [bilyu] (white), *сонячний* [sonyachnyu] (sunny), *космічний* [sonyachnyu] (cosmic). But the mood is not a feature of the language and depends on the meaning of the words/phrases according to the surrounding context of the text, for example, the word *кумедний* [kumednyu] (funny) has several interpretations of conveying the mood, in particular, positive – *смішний клоун* [smishnyu kloun] (funny clown), negative – *кумедний одяг* [kumednyu odyah] (funny clothes), and neutral – *кумедний кіт* [kumednyu kit] (funny cat) or *кумедна іграшка* [kumedna ihrashka] (funny toy). The word *гострий* [hostryu] (sharp) from the word *перець* [perets'] (pepper) or *ніж* [nizh] (knife) has a positive meaning when buying, but from the word *біль* [bil'] (pain) and *ніж* [nizh] (knife) in a criminal case, it has a negative meaning. Also, negation turns the meaning of a positive text with positive words into a negative one and vice versa, for example, *ми дуже багато очікували від відпочинку на морі сонячними гарними днями, але обіцяна курортна база відпочинку все спаскудила* [my duzhe bahato ochikuvaly vid vidpochynku na mori sonyachnymu harnymu dnyamy, ale obitsyana kurortna baza vidpochynku vse spaskudyla] (we expected a lot from a vacation at the sea on sunny, beautiful days, but the promised holiday resort spoiled everything) (one negative word *спаскудила* [spaskudyla] (spoiled) all the previous positive ones) or *дощ, прохолода та вітер не стали перепонами гарно відпочити в чудовій компанії* [doshch, prokholoda ta viter ne staly pereponamy harno vidpochyty v chudoviy kompaniyi doshch, prokholoda ta viter ne staly pereponamy harno vidpochyty v chudoviy kompaniyi] (rain, coolness and the wind did not become an obstacle to a good rest in a wonderful company). Only thanks to machine learning in such cases it is possible to get the

predictability of the text and reveal the emotional colouring according to the context. An a priori deterministic/structural approach loses the flexibility of context and meaning, so most speech models take into account the location of words in context, using ML methods for prediction. The main method of developing simple speech models is the bag of words as the frequency of co-occurrence of words in a narrow, limited context (Fig. 4).



**Figure 4:** Frequency matrix of co-occurrence of words

Such evaluation helps to determine the probability neighbourhood and to determine their meaning from small fragments of text. Next, using statistical inference methods, word order can be predicted. This is quite simple for English texts where words are not inflected. For Ukrainian language tests, it is better to use not a bag of words, but a bag of word bases. For example, for 12-word combinations as a 3-gram (36 words) without taking into account declension, we will get a matrix of size 20×20, and with consideration of declension, gender and person (analysis of only the bases of words) – 15×15. Moreover, for the Ukrainian language, the location of bases in the 3-gram is usually not important and often has an unambiguous probability of compatibility in terms of content, for example, *інформаційний ресурс* (*інформ ресурс*) [informatsiynyu resurs (inform resurs)] (information resource

(inform resource)) and *ресурс інформації* (*ресурс інформ*) [resurs informatsiyi (resurs inform)] (information resource (inform resource)). The bag-of-words/stems model is also extended by analyzing the co-occurrence of stable phrases and fragments of expressions that are of great importance for identifying the meaning of the text. The expressions *зелений край скатертини* (*межа*) [zelenyy kray skatertyny (mezha)] (green edge of the tablecloth (border)) and *зелений край батьківщини* (*місцевість*) [zelenyy kray bat'kivshchyny (mistsevist')] (green edge of the homeland (locality)) in the form of a 3-gram carry a different meaning. That is, there are several interpretations only for the word edge (the boundary of an object, a piece, the end of an action/state, a special area, a place of residence, an administrative-territorial unit). Statistical analysis of n-grams makes it possible to distinguish patterns of context. Speech models based on the analysis of n-gram contexts require the ability to explore the relationship of text to some target variable. The application of the analysis of linguistic and contextual features contributes to the formation of the general predictability of the text. However, their identification and further use require the ability to parse/identify the linguistic units of the language.

3. An example of the analysis of a structural feature can be the construction of an ontology for the implementation of IIS. Along with linguistic and contextual features, it is then necessary to identify and process high-level language units to define a vocabulary of operations for the text corpus. Different units of language are processed at different levels, and the correct implementation of NLP methods based on ML is important for the operational and correct identification of the linguistic context (sem relationship structure). Based on a typical pattern of utterances (statement or simple phrase) in the form of the *subject* → *verb* → *object* → *object definition* (*subject* → *predicate* → *appendix*) construct ontologies that define specific relationships between entities. They make it possible to solve the problem of the lack of a mandatory order of words in a Ukrainian sentence to identify its semantics. It is advisable to use it for tasks where it is necessary to constantly process large volumes of text data and there is long-term resource support for the project. Semantic analysis consists not only in identifying the content of the text but also in generating data structures to which logical reasoning can be applied. Thematic Meaning Representations (TMR) are used to encode sentences in the form of predicate structures based on first-order logic or lambda calculus ( $\lambda$ -calculus). Network/graph structures are used to encode interactions of predicates of relevant text features. Then a traversal is implemented to analyze the centrality of terms or subjects and the reasons for the relationships between elements. Graph analysis is usually not a complete SEM (semantical analysis), but helps to form part of important logical decisions or conclusions. Semantics, syntax and morphology allow you to add data to simple text strings with linguistic meaning and generate new meaningful text content. Nowadays, natural language is one of the most commonly used forms of content. Its analysis makes it possible to increase the usefulness of data applications and make them an integral part of everyday life. Scalable analysis and machine learning of text primarily require up-to-date knowledge and text corpora of the relevant SA. For example, in the field of finance, CLS needs to identify financial terms, stock abbreviations and company names. Therefore, documents in the SA corpus must contain these entities. That is, the development of any CLS begins with obtaining textual data of the appropriate type and forming a corpus with structural and contextual features of SA.



## 4. Experiments, results and discussions

### 4.1. Method of grapheme analysis of the Ukrainian language

For the GA of text strings, it is best to use regular expressions (RE) as algebraic notations for the features of a set of character strings. Commonly used in the development/maintenance of each type of computer language (programming, communication protocols, data markup, specification, and design), the operation of text editors, and word processing software, especially with IIS templated or SA text corpora collections. Identification/search of a fragment/string by pattern in a sequence of character strings is implemented to find all matches or the first one. The templates use special characters [ , ^, \, -, \*, +, ., \$, |, (, ), \_ {, } ], etc., including /, but the latter is not RE, but its boundaries. The simplest RE is a tuple of simple characters (Table 1) to recognize the first or all pattern-like occurrences of character sequences.

**Table 1**

Regular expressions of GA texts in the Ukrainian language for recognition of all characters

N	RE	Recognition	Example and result
1	/контент/	the exact sequence of substring characters, taking into account the case	Структурна схема лінгвістичного аналізу текстового <u>контенту</u>
2	/к/	a specific character, taking into account the case	Контент-аналіз застосовують для аналізу потоків <u>контенту</u>
3	/-/	specific special character	Контент-аналіз застосовують
4	/[кК]онтент/	exact sequence of characters without taking into account the case of the 1st character	<u>Контент</u> -аналіз застосовують для аналізу потоків <u>контенту</u>
5	/[онві]/	or o, or н, or в, or і	<u>Контент-аналіз</u> застосовують
6	/[0123456789]/	Any number in a string sequence	RE чутливі до регістру– правила <u>1, 2</u> та <u>4</u> дають різні результати
7	/[0123]/	or 0, or 1, or 2, or 3	RE чутливі до регістру– правила <u>1, 2</u> та <u>4</u> дають різні результати
8	/[0-9]/	Any number in a string sequence	RE чутливі до регістру– правила <u>1, 2</u> та <u>4</u> дають різні результати
9	/[а-я]/	Any lowercase letter of the Ukrainian alphabet	<u>Контент-аналіз застосовують</u>
10	/[А-Я]/	Any uppercase letter of the Ukrainian alphabet	<u>Контент-аналіз застосовують</u>
11	/[А-Яа-я]/	Any letter of the Ukrainian alphabet, regardless of case	<u>Контент-аналіз застосовують</u>
12	/[A-Z]/	Any uppercase letter of the English alphabet	<u>RE</u> чутливі до регістру– правила 1, 2 та 4 дають різні результати
13	/[^А-Я]/	Any character other than an uppercase letter of the English alphabet	<u>Контент-аналіз застосовують для аналізу потоків контенту</u>
14	/[^кК]/	Any character except the letters К and к	<u>Контент-аналіз застосовують для аналізу потоків контенту</u>
15	/[^\.]/	Any character except the dot character.	<u>Контент-аналіз застосовують</u>
16	/[к^]/	or к, or ^	аналіз потоків <u>контенту</u>
17	/x^y/	String pattern x^y	функція <u>x^y</u>
18	/^[А-Я]/	Any uppercase letter of the Ukrainian alphabet at the beginning of a line	<u>Контент-аналіз застосовують для аналізу потоків контенту в CLS</u>
19	/^а/	The letter a at the beginning of the line	Контент-аналіз застосовують

N	RE	Recognition	Example and result
20	/контенту?/	Presence/absence of the optional y character in the substring	Структурна схема лінгвістичного аналізу текстового <b>контенту</b>
21	/зв'язок/	The apostrophe ' is optional for searching and is often omitted	Структурна ознака описує зв'язок між лінгвістичними лексемами.
22	/лін.вітиска/	Designation of any symbol	лінґвістика або лінґвістика
23	/б.гу/	Designation of any symbol	Зараз змагання з <b>бігу</b> , тому я <b>біжу</b> . Я <b>бігун</b> , тому <b>біжу</b> естафету
24	/i*/	Any line without i or any number of i	<b>МА лексеми провадять на основі її особистої множини ознак</b>
25	/ii*/	Any string with one or more i	МА лексеми провадять на основі <b>ii</b> особистої множини ознак
26	/[нжтлдчз]*/	or without or any number of н or ж or т or л or д or ч or з	<b>Віддалено летється на ланах нашого життя беззмінне збіжжя знання як обличчя особистого досвіду!</b>
27	/[нжтлдчз]/	or н or ж or т or л or д or ч or з	<b>Віддалено летється на ланах нашого життя беззмінне збіжжя знання як обличчя особистого досвіду!</b>
28	/[0-9]*/	or none or an arbitrary number of one element from the range 0-9	<b>RE чутливі до регістру– правила 1, 2 та 4 дають різні результати</b>
29	/[0-9][0-9]*/	One digit from the range 0-9 is required, the other is not, but if there is - any number of one of 0-9	RE чутливі до регістру– правила <b>1, 2</b> та <b>4</b> дають різні результати
30	/[0-9]+/	any number of different digits from 0-9	Спецсимвол знаку питання ? для RE-правил <b>20-21</b>
31	/[нжтлдчз]+/	one or н or ж or т or л or д or ч or з or several, or any combination thereof	<b>Віддалено летється на ланах нашого життя беззмінне збіжжя знання як обличчя особистого досвіду!</b>
32	/[нжтлдчз]{2}/	exactly two or н or ж or т or л or д or ч or з	<b>Віддалено летється на ланах нашого життя беззмінне збіжжя знання як обличчя особистого досвіду!</b>
33	/аналіз.*аналіз/	String identification using a double word <b>аналіз</b>	Контент- <b>аналіз</b> застосовують для <b>аналізу</b> потоків контенту в CLS
34	/^В/	<b>В</b> at the beginning of the line	<b>В</b> наш час в Інтернет все є.
35	/^Контент-аналіз\$/	recognition of a specific phrase	<b>Контент-аналіз_</b>
36	/_\$/	marking a space at the end of a line	Контент-аналіз_ застосовують_
37	/^Контент-аналіз\._\$/	recognition of a specific phrase with a period and a space at the end of the line	<b>Контент-аналіз_.</b>
38	/^[А-Я]\._\$/	recognition of all possible sentences	<b>В наш час в Інтернет все є_.</b>
39	/\баналіз\b/	recognition of a specific set of symbols (words) taking into account boundaries	Контент- <b>аналіз</b> застосовують для аналізу потоків контенту
40	/\b19\b/	recognizing a word as a number	Йому виповнилось <b>19</b> в 2019.
41	/\b3\b/	word recognition within limits	Ціна - <b>3</b> \$ за 13 одиниць.
42	/\b5\b/	word recognition within limits	Ціна -5Є за <b>5</b> одиниць.
43	/ML MH/	recognition of abbreviations ML or MH	Реалізація CLS на основі <b>ML</b>
44	/контент(у ний)/	recognition of words with different inflections	<b>Контентний</b> аналіз застосовують до великих потоків <b>контенту</b>
45	/№_[0-9]+_*/	1 digit with any number of spaces	В_колонці <b>№ 3</b>
46	/((№_[1-9]+_*)*/	recognition of arbitrary sequence number <b>№</b> and any number	В_колонках <b>№ 1</b> та <b>№ 3</b> , але не в <b>№_13_</b>

RE is case-sensitive – rules 1, 2 and 4 give different results. Using the special characters [ and ] solves the case-sensitivity problem of RE. The string of characters in the middle of []

implements the disjunction of the values upon matching. RE-rule 6 recognizes any number in a sequence of string characters. The dash special character - in the middle [] for RE-rules 8-12 allows not to list all characters but indicates any character in the corresponding range. For example, Pattern /[3-6]/ indicates any of the characters 3, 4, 5, or 6, and /[в-ж]/ indicates one of the characters в, г, д, or ж in the grapheme analysis of the input test. The caret or circumflex character ^ inside [] for RE-rules 13-18 carries a different content load depending on the location. If at the beginning immediately after [ means, all characters after it are rejected in the parsed character string (RE 13-15). The caret ^ has 3 purposes: to indicate the beginning of a line (not inside [] - RE 18-19); to indicate negation within [] (RE 13-15); simply to denote carriage ^ (RE 16-17). Question mark special character ? for RE-rules 20-21 allows you to mark optional characters in the searched string. This is useful in cases where there may be both present/absent characters in a certain sequence that do not resolve []. In [] - you can indicate the absence of a specific symbol from the range of possible ones, but do not describe the absence of any symbol at all, indistinguishable from ?. The dot special character . for RE-rules 22-23 allows you to mark the location of any symbol in the sequence of the analyzed string. If the special character ? there is the absence or presence of one symbol, then we can submit the doubling of the symbol through the special symbol \* (RE 26-29), which means the absence of a specific symbol or RE before \* in the RE or its arbitrary number in consecutively placed in the recognized line, i.e. the result can be a line without this symbol. Therefore, to find at least one symbol from a possible sequence of the same two - for example RE 29, and for two different ones - 30. The + special character for RE-rules 30-31 allows you to mark one or more cases immediately preceding the /RE symbol. {} (RE 32) is used to indicate the exact quantity (for example, exactly 2 times). The dot special character. often used together with the special character \* to indicate any string of characters (RE 33).

An anchor is a special symbol (for example, a double sign ^ or a dollar sign \$) specifying the location of the RE in the character string. In some cases, the caret ^ marks the beginning of a line (RE 34). The dollar sign \$ recognizes the end of a line (RE 35-36). The backslash \ allows you to recognize special characters in the character string of the input test (RE 37-38). The anchors \b and \B identify the presence and absence of word boundaries, respectively (RE 39-42). A word is any tuple of numbers, underscores or letters (without special characters).

To organize the selection of alternatives between, for example, synonyms, the disjunction operation based on the special symbol | (RE 43-46). The combination of special characters | inside () allows you to arrange disjunction recognition only for a specific pattern, taking into account different inflexions/prefixes (RE 44). Special characters () are used to organize counters of type \* (RE 46). The difference is that \* is used for one character, not a whole sequence.

For complex disjunctive RE operators, when grouping from different special symbols, the concept of priority is used (Table 2): () → \*, +, ?, {} → string, ^, \$ → | from the highest to the lowest (delimitation by the symbol →) (). Greedy RE patterns of the type /[a-ya]\*/ recognize zero or more letters and no matches, expanding the identification to cover as many strings as they can. Non-greedy RE based on \*? and +? find the smallest possible text. RE of the type /\_\*/ is used to indicate the absence or presence of a certain number of spaces,

since there can always be additional spaces around. There are aliases for general ranges that can be used primarily to preserve grapheme type (Table 3). Correctly constructed REs avoid errors of assumption (overrecognition) and negation (accidentally missed). Reducing the overall error rate for GA implies two antagonistic conditions for generating a collection of REs increasing recall (minimizing false ignores) and increasing precision (minimizing false recognitions).

**Table 2**

Regular expressions to recognize keywords, stop words and tokens

N	RE	Recognition
1	/але/ /аналіз/	simple (but incorrect) pattern - takes into account other possible variants of the sequence of characters in the input string
2	/[aA]ле/ /[aA]наліз/	from case-sensitive, but unfortunately takes other cases into account, such as <i>мале</i> ча or <i>каналізація</i>
3	/\b[aA]ле\b/ /\b[aA]наліз\b/	taking into account the boundaries of the word (without letters, underscores and numbers on both sides) - for but good, but the word analysis already ignores
4	/[^\a-яА-Я][aA]наліз[a-я]/	Before <b>аналіз</b> there was not a single letter regardless of case, and after it is an arbitrary lowercase letter of the Ukrainian alphabet
5	/\b[aA]наліз[a-я]*/	Before <b>аналіз</b> there is no letter, underscore or number, followed by any lowercase letter of the Ukrainian alphabet or none
6	/(^\ b[aA]ле\b/ /(^\ b[aA]наліз([a-я]* \$/	to item 5, the possibility of meeting the word analysis at the beginning or the end of the line is added, when no character exists in these positions
7	/[0-9]+ (\\$ грн\. EU)/	the integer value of the price in грн. (UAH), or US/EU currency
8	/[0-9]+\.[0-9][0-9] грн\./	the actual value of the price in грн. (UAH)
9	/(^\ W)[0-9]+\.[0-9][0-9]? (\\$ грн\. EU)?\b/	the actual value of the price in the currency of Ukraine/USA/EU at the level of a word in a sentence/utterance/phrase
10	/(^\ W)[0-9]{0,5}(\.[0-9][0-9])? (\\$ грн\. EU)?\b/	the actual value of the price in the currency of Ukraine/USA/EU at the word level, taking into account the limitation of the number of digits before the comma
11	/\b[6-9]+_*(UAH € грн\. [Гг]рив(ня ні ень))\b/	lines with a price value > 5 in the currency of Ukraine, taking into account various options for designations and abbreviations
12	/\b[0-9]+\.[0-9]+)?_*(UAH € грн\. ?)\b/	lines with the valid value of the price in the currency of Ukraine, taking into account the presence/absence of various options for designations and abbreviations

**Table 3**

Basic RE aliases for general GA ranges

N	Range	RE	Recognition	Example
1	[\_n\t\r]	\s	any spaces and tabs	аналіз_контенту
2	[^\s]	\S	no spaces or tabs	<u>а</u> наліз_контенту
3	[0-9]	\d	any number from the range	<u>14</u> _лютого_2005
4	^[0-9]	\D	no digit from the range	14_ <u>л</u> ютого_2005
5	[a-яА-Я0-9_]	\w	any letter, number and underscore	<u>к</u> онтент_аналіз
6	[^\w]	\W	no letter, number or underscore	контент_ <u>а</u> наліз
7	\b[0-9]*\b	*	none or several previous REs	<u>вже 22 рік</u>
8	\b[0-9]+\b	+	one or more previous RE	вже <u>2022</u> рік
9	\b[0-9]?\b	?	definitely absent or present once	<u>22 рік 2 століття</u>
10	\b[0-9]{2}\b	{n}	a certain number of repetitions	<u>22</u> рік 2 тисячоліття
11	\b[0-9]{1,2}\b	{n,m}	in the range of a certain number of repetitions	<u>22</u> рік <u>2</u> тисячоліття
12	\b[0-9]{2,}\b	{n,}	at least a certain number of repetitions	<u>22</u> рік 2 тисячоліття
13	\b[0-9]{,2}\b	{,m}	to a certain number of repetitions	22 рік <u>2</u> тисячоліття

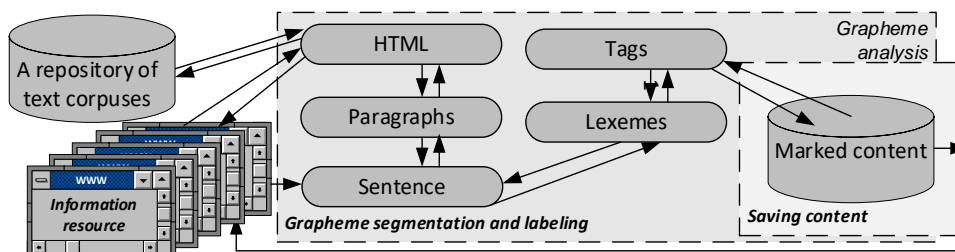
N	Range	RE	Recognition	Example
14	[0-9]{1,}\*[0-9]{1,}	\*	special character designation *	значения <b>5*93</b>
15	1[0-9]{1}\.0[0-9]{1}	\.	special notation for the dot sign	дата <b>14.02</b>
16	[a-я]\?	\?	special designation of the question mark	контент-аналі <b>з?</b>
17	[a-я]\n[a-я]	\n	special notation for the newline character	контент-аналі <b>з</b> контент- моніторинг а) <b>б) с)</b>
18	[б-я])\t[a-я])	\t	special notation for the tab character	текст → контент
19	s/текст/контент/	s/x/y/	replacement/clarification of the word by another	27 → <27>
20	s/([0-9]+)/<\1>/	s/R/R'/	expression replacement/clarification with a template	x <b>A</b> y <b>A</b> z
21	/x(*)y\1z/	/(.*)\1/	repeating a certain line/expression twice	/x <b>A</b> y <b>BzAwBu</b> /
22	/x(*)y(*)z\1w\2u/	/()()\1\2/	duplicates of two expressions in certain places	<b>x w</b> text <b>x w</b>
23	/(?:x y)(z )text	/(?: )( )	grouping, without fixing the template	контент-аналі <b>з</b>
24	/(?![яЯ]) [A-Яa-я]+/	/(?!x)y/	any string that does not begin with я	

RE  $/\{9\}/$  is the recognition of exactly 9 cases of the previous symbol/expression, RE  $/a.\{3\}я/$  – sequences v, RE  $/\{3,12\}/$  – from 3 to 12 of the previous symbol/expression, RE  $/\{5,\}/$  is at least 5 occurrences of the preceding character/expression, and RE  $/\{,13\}/$  is up to 13 occurrences of the preceding character/expression. The special character **s** before RE allow you to replace the expression with a pattern. The special character  $\backslash k$  indicates the location of the character/phrase/expression as a duplicate of the first element in the capture group, i.e. the pattern in  $()$ , where  $k$  is the number of brackets or capture groups. Thus, special characters  $()$  have a double function in RE: to group conditions and to determine the order of application of operators. For grouping, without fixing the received template in the register, the RE of the form  $(?: \text{template})$  is used as a group that does not capture the expression. When applying RE, the rank of use in the queue is determined. An RE of type  $(?: \text{template})$  is a positive statement (RE 23).

The  $(?= \text{pattern})$  operator is positive when identifying a zero-width pattern, i.e. the match pointer is not advanced. The  $(?! \text{pattern})$  operator is positive if the pattern does not match, is zero-width, and the cursor does not advance. Negative statements are usually used in the analysis of a complex content model when a special case needs to be removed (RE 24).

Grapheme analysis is the preliminary processing and transformation of the text into a certain marked and compressed format for the following NLP processes (Fig. 5):

*extracting content* → *extracting paragraphs* → *extracting sentences within a paragraph* → *extracting tokens within a sentence* → *marking tokens with tags for MA as part-of-speech marking*.



**Figure 5:** Content partitioning, grapheme segmentation and labelling

At the first stages of the integration of content from various sources, it is necessary to implement the processes of filtering, access and calculation of text sizes based on the application of the standard API of pre-grapheme processing of the division of documents through the execution of the following sequence of NLTK methods:

- $f_{raw}()$  is organization of access to previously unprocessed text;
  - $f_{\text{html}}()$  is the elimination of non-text content, scripts and style tags;
  - $f_{patas}()$  is the identification of individual paragraphs from the content text;
  - $f_{sents}()$  is the identification of individual sentences from the content text;
  - $f_{tokens}()$  is the identification of individual tokens from the content text;
  - $f_{mark}()$  is grapheme labelling of identified tokens based on RE;
- $$T_{\text{marked}} = f_{\text{mark}}(f_{\text{tokens}}(f_{\text{sents}}(f_{\text{patas}}(f_{\text{html}}(f_{\text{raw}}(X_{\text{content}})))))), \quad (6)$$

and if necessary, additional methods, such as adding tags or parsing sentences, converting annotated text into tree-like data structures, or extracting individual XML elements. To identify and extract the main content from an information resource with an undefined structure and high variability of documents from different sources,  $f_{\text{html}}()$  based on the Python readability-lxml library is used, which removes all anomalous artefacts, leaving only the text. When processing HTML text,  $f_{\text{html}}()$  uses a collection of formal REs to identify and remove navigation menus, declarations, script tags, and CSS, then creates a new content object model tree, extracts the text from the source tree, and embeds it into the newly created tree.

Vectorization, feature extraction, and ML tasks rely heavily on CLS's ability to efficiently break down textual content into its constituent components while preserving the original structure. The accuracy and sensitivity of ML models depend on the efficiency of identifying the connections of tokens with the corresponding context in the text. Paragraphs contain complete ideas of context and are the structural unit of content. Based on NLTK, the  $f_{patas}()$  operator is implemented as a paragraph generator, which is defined as blocks of text separated by two newline characters. The  $f_{patas}()$  the operator scans all files and passes each HTML text to the RE constructor, indicating that parsing of the HTML markup should be done through the lxml HTMLparser. The resulting object maintains a tree structure that can be navigated using native HTML tags and elements.

If paragraphs are structural units of content, then sentences are semantic units. As a paragraph expressing a single idea, a sentence contains a complete thought that the author has formulated and expressed in many words. Grapheme segmentation is the division of text into sentences for further processing by marking words with parts of speech in MA. The operator  $f_{sents}()$ , calling  $f_{patas}()$  and returning an iterator (generator), sorts all sentences from all paragraphs.

The  $f_{sents}()$  operator bypasses all paragraphs selected by the  $f_{patas}()$  operator and uses the  $f_{words}()$  operator to perform the actual grapheme segmentation. Internally, the  $f_{tokens}()$  operator uses  $f_{mark}()$ , a model pre-trained with RE recognition/identification rules for various kinds of tokens, punctuation marks, abbreviations, geographical names, abbreviations, and other marks that serve as sentence start/end or tab marks. Punctuation marks do not always have an unambiguous interpretation, for example, or are a sign of the end of a sentence, but they are also present in dates, abbreviations, abbreviations, ellipses,

etc. Determining sentence boundaries is not always an easy task. Punctuation is crucial for identifying word boundaries (commas, spaces, colons) and for identifying certain aspects of meaning (question marks, exclamation marks, quotation marks). For some tasks, such as tagging parts of speech, and analyzing or synthesizing speech, it is sometimes necessary to treat punctuation marks as if they were separate words. When analyzing speech, punctuation marks replace pauses, accents, and changes in intonation dynamics. Lexemization is the process of obtaining lexemes (syntactically encoded strings of symbols) and for its implementation, the operator  $f_{words}()$  based on RE is used, which is selected through  $f_{mark}()$  markers for spaces and punctuation marks and returns a list of alphabetic and non-alphabetic characters. Like delimiting sentences, lexeme recognition is not always an easy task: the presence of punctuation marks in a lexeme, punctuation marks as independent lexemes, lexemes with and without hyphens, and lexemes as shortened forms of words (one or more words). Different marker selection tools are chosen for these cases. Any statement is a speech correlate of a sentence. The presence of lexemes of the dysfluency type (loss of speech speed, for example, a longer pause when thinking) carries not so much a semantic load as an emotional one. Exclamations such as *mmmm, oh, ah* [mmmm, ohh, ah], etc. are fillers or filled pauses and are also emotionally coloured, but not semantically coloured. An unfinished word with further repetition and its ending or simply with repetition is a fragment that does not carry a semantic load, but only an emotional one. Therefore, when conducting PHA, depending on the goal of solving a specific problem through CLS, it is important to take into account (mark accordingly) or ignore some types of punctuation (ellipsis, exclamation points, etc.), dysfluencies, double fragments, exclamations, etc. If CLS is just a transcription of speech, then such phonemes should be ignored to avoid loss of speech rate. But they make it possible to determine the psychological state of the speaker and his emotional state, to identify the peculiarity of the speaker's authorial speech when the tone of the voice changes, they are relevant in predicting the future word, because they signal that the speaker is restarting the statement/idea, and therefore, for speech recognition, ordinary tokens are considered as phonemes. Marking a lexeme as a lemma (a set of lexical forms having the same base, the same main part of speech and the same word content) or as a word form (a fully inflected or derived form of a word) is a significant difference for conducting the next stage of MA as lemmatization or stemming, i.e. identification of word bases. For many NLP tasks in the English language, it is enough to mark the corresponding lexemes as word forms, but for the Ukrainian language – no, it is still necessary to identify the bases of the words (for example, based on the analysis of inflexion according to the tree of endings).

There are two ways to identify words with punctuation ignored - token recognition as types (the number of different words  $|V|$  in the set of words of the corpus, i.e. the cardinality of the alphabet of the corpus, where an element of the alphabet/dictionary is a unique word) and tokens (the total number  $N$  of words of the analyzed corpus), i.e.  $|V| \leq N$ . The largest Google N-grams corpus contains 13 million types among those displayed  $\geq 40$ , so the true number is much larger.

The ratio between the number of types  $|V|$  and the number of tokens  $N$  is called **Herdan's law** (Herdan, 1960) or **Heaps' law** (Heaps, 1978):  $|V| = kN^x$ , where  $k$  and  $x$  are positive constants for  $0 < x < 1$ . The value of  $x$  depends on the size of the corpus and the genre, for

large corpora  $x$  varies within [0.67; 0.75], when the size of the dictionary for the text grows much faster than the square root of the length of its words. Another measure of the number of words in a language is the number of lemmas rather than word types (for example, the Oxford English Dictionary has over 615,000 entries).

#### 4.2. Method of morphological analysis of the Ukrainian language

Morphology identifies the shape of things, and in textual analysis, the shape of individual words/tokens. Lexemes are both words and punctuation marks, allowing you to conduct the next SYA (syntactic analysis) more clearly. Word structure helps determine plural, gender, tense, person, declension, etc. MA is a difficult task, as most languages have many exceptions to the rules and special cases. The main task of MA is to identify parts of words to assign them to certain classes (tags) of parts of speech. For example, sometimes it is important to understand whether a noun is singular or plural, or is a proper name. It is also often necessary to know whether the verb has an indefinite form, past tense, or is an adjective. The resulting parts of speech are then used to generate larger structures (fragments/phrases), or whole word trees, which are then used to build semantic reasoning data structures. After GA (grapheme analysis), we have access to tokens in sentences in paragraphs of integrated content texts, which makes it possible to apply MA to mark words from the collection of tokens with parts of speech (e.g., verbs, nouns, prepositions, adjectives) that indicate the role of the word in the context of the sentence. In the Ukrainian language, the same word can usually take on different roles, depending on the inflexions. Part-of-speech tagging based on MA rules consists of adding a corresponding tag to each word from a collection of tokens that contains information about the definition of the word and its role in the current context. MA rules are used for the development of modules/subsystems for keyword identification, text classification (Fig. 4.6), machine translation, and error correction, as well as for human psychological analysis, semantic analysis, etc. When identifying words for further classification, the `rub_id` attribute describes the rubric to which a specific keyword belongs (Table 4).

**Table 4**

Examples of Ukrainian and English words/flags for identifying keywords

N	Ukrainian	English	N	Ukrainian	English
1	курсорний/V	cursoriness/17,13	39	буферизувати/ABGH	buffer/18,9,13,17,10,23
2		cursorily	40	відформатувати/AB	format/1,20,17
3		cursor/9,13,17,10	41	кодувати/ABGH	code/17,2,23,10,12,18,9
4		cursorily/16	42	кешувати/ABGH	cache/9,17,18,10,13
5	кириличний/V	Cyrillic	43	кука/ab	hook/10,23,9,18,13,17
6	кілобітовий/V	kilobit/17	44	клавіатурний/V	keyboard/18,9,13,23,10,17
7	кілобіт/efg		45	клавіатура/ab	
8	кілобайтовий/V	kilobyte/17	46	кодосумісний/V	code/17,2,23,10,12,18,9
9	кілобайт/efg		47		code compatible
10	кодек/efg	coder/2,13	48		compatible/17,5
11	кодер/efg		49		compatibleness/13
12	консольний/V	consoled/7	50		compatibility/5,13,17
13		consoler/13	51		compatibly/5
14	консоль/ij	console/23,8,10	52	кодогенератор/efg	code/17,2,23,10,12,18,9
15	Кобол/e	COBOL	53		generators/1



N	Ukrainian	English	N	Ukrainian	English
16		Cobol/13	54		generator/17,13
17	кілобод/efg	kilobaud/13	55	конфігуратор/efg	configuration/1,17,13
18	копілефт/е	Copyleft/19,18,17	56		configure/1,10,17,9,8
19	хакер/efg	hacker/13	57	криптозахиснений/V	crypto-protected/7,21
20	хеш/е	hash/1,10,17,9	58	криптографічний/V	cryptographic
21	таймер/efg	timer/13	59		cryptographically
22	стек/efgo	stack/13	60		cryptography/13,17
23	спам/е	spam/13	61	копірайт/е	copyright/13,17,18,9,10,23
24	смайл/ef	smile/10,13,9,17,18	62	комутований/V	switch/10,8,23,13,18,17,9,12
25	сайт/ef	site/9,17,12,13	63	конкатенація/ab	concatenate/22,17,9,10
26	рестарт/ef	restart/8	64	комбосписок/ab	combo/13,17
27	рекурсія/ab	recursion/13	65		box/9,18,17,12,23,10,13
28	процесор/efg	processor/13,17	66		list/12,13,18,9,15,10,23,22,17
29	проксі	proxy/17,13	67	крос-компілятор/efg	cross/13
30	принтер/efg	printer/1,13	68		compilable/7
31	подкаст/е	podcast/13	69		compilation/17,1,13
32	плотер/efg	plotter/13,9,17,10	70		compile/1,17,9,2,10
33	піксель/efg	pixel/17,13	71		compiler/2,17
34	опція/ab	option/10,9,13,17	72		compiler's
35	офлайн/е	offline/13	73	крос-асемблер/efg	cross-assembler/3,13,17
36	онлайн/е	online/13	74	фрейм/efg	frame/17,18,9,12,10,13,23
37	модем/efg	modem/17,13	75	файл/ef	file/6,9,18,17,10,13,23
38	сплайн/efg	spline/13,17,9	76	сигнатура/ab	signature/13,17

The flag of the attribute defines the properties of this keyword (the part of the language to which it belongs). In thematic dictionaries, each word has its property, for example, a b c d o – different types of nouns, A – verbs, V – adjectives (Fig. 7). To compare the complexity in thematic dictionaries (23 rules in total), each English word also has a property, for example, the numbers 1-23 are the numbers of rules of the PFX type (prefixes, rules 1-7) and SFX (suffixes and endings, rules 8-23) and describe some nouns for English words (Fig. 8). For example, PFX-type rules describe the modification of some nouns for English words with prefixes: re- (rule PFX 1), de- (rule PFX 2), dis- (rule PFX 3), con- (rule PFX 4), in- (PFX rule 5), pro- (PFX rule 6) and un- (PFX rule 7).

id	ordering	state	word	rub_id	flag	lang	params	attribs	language
70597	0	1	підтвердження	4	ij	uk			*
71207	0	1	планшет	3	efg	uk			*
71209	0	1	план	4	ef	uk			*
71728	0	1	побічний	2	VW	uk			*
72303	0	1	повнота	2	aZ	uk			*
72555	0	1	погодження	3	ijZ	uk			*
74132	0	1	поліморфний	2	V	uk			*
74194	0	1	політехніка	0	ab	uk			*
74195	0	1	політехнікум	0	ef	uk			*
74196	0	1	політехнічний	0	V	uk			*
74208	0	1	політико-економічний	0	V	uk			*
74209	0	1	політико-правовий	0	V	uk			*
74510	0	1	помилка	0	ab	uk			*
74604	0	1	помножений	0	VW	uk			*
75979	0	1	порівняльний	0	V	uk			*
76540	0	1	послідовний	0	VWZ	uk			*
77892	0	1	предмет	0	efg	uk			*
77927	0	1	презентація	0	ab	uk			*
77928	0	1	презентований	0	VW	uk			*
77929	0	1	презентувати	0	ABGH	uk			*
81186	0	1	промислово-економічний	0	V	uk			*
81187	0	1	промислово-інвестиційний	0	V	uk			*
81188	0	1	промислово-фінансовий	0	V	uk			*
81435	0	1	пропорція	0	ab	uk			*
82223	0	1	професор	0	efg	uk			*

Figure 6: The relation of keywords in the CLS database of text rubrics

```

# Групи а в с д о
#
# -- Перша відміна: іменники жіночого та чоловічого та середнього роду
#
# -- Друга відміна: іменники чоловічого роду із закінченням на -ар -ир
#                    наголошені (мішана група на -ар -ир)
#
# -- Друга відміна: іменники чоловічого роду з чергуванням -і -о
#
# -- Числівники -ять, -сят, -сто
#
#
SFX а Y 235

#
# ОДНИНА (множина перенесена в гр. б)
#
# Спочатку перша відміна
#
# тверда група в Називному відмінку однини з закінченням на -а
# однина
SFX а а и [^жщ]а # хата хати (Р.)
SFX а а і [^ггкх]а # хата хати (Д.М.)
SFX а а у а # хата хату (3.)
SFX а а ою [^жщ]а # хата хатою (0.)

```

Figure 7: Noun classification dictionaries for Ukrainian words

TRY esianrto1cdugmphbyfvkwzESIANRTO1CDUGMPHBYFVKWZ'оаитенрсвилпкыяудмзшбчгцюжцѣхфѣАВСМКГПТЕИЛФНДОЭРЭЮЯБХЩЦУЧЬЫЩЙЇ

FLAG num

NOSUGGEST 49

ONLYINCOMPOUND 50

COMPOUNDMIN 1

REP 88

REP а ei

REP ei а

REP а ey

REP ey а

REP ai ie

REP ie ai

REP are air

REP are ear

REP are eir

REP air are

REP air ere

REP ear air	^ SFX 9 Y 4	^ SFX 16 0 ness [^y]
REP air ear	SFX 9 0 d e	SFX 17 Y 4
REP w qu	SFX 9 y ied [^aeiouy]	SFX 17 y ies [^aeiouy]
REP qu w	SFX 9 0 ed [^ey]	SFX 17 0 s [aeiouy]
REP z ss	SFX 9 0 ed [aeiouy]	SFX 17 0 es [sxzh]
REP ss z		SFX 17 0 s [^sxzhy]
REP shun tion	SFX 10 Y 2	SFX 18 Y 4
REP shun sion	SFX 10 e ing e	SFX 18 0 r e
REP shun cion	SFX 10 0 ing [^e]	SFX 18 y ier [^aeiouy]
		SFX 18 0 er [aeiouy]
		SFX 18 0 er [^ey]
PFX 1 Y 1	SFX 11 N 2	SFX 19 N 4
PFX 1 0 re .	SFX 11 y ieth y	SFX 19 0 st e
	SFX 11 0 th [^y]	SFX 19 y iest [^aeiouy]
		SFX 19 0 est [aeiouy]
		SFX 19 0 est [^ey]
PFX 2 Y 1	SFX 12 Y 2	SFX 20 N 2
PFX 2 0 de .	SFX 12 e ings e	SFX 20 e ive e
	SFX 12 0 ings [^e]	SFX 20 0 ive [^e]
PFX 3 Y 1	SFX 13 Y 1	SFX 21 Y 1
PFX 3 0 dis .	SFX 13 0 's .	SFX 21 0 ly .
PFX 4 Y 1	SFX 14 Y 1	SFX 22 Y 3
PFX 4 0 con .	SFX 14 0 ment .	SFX 22 e ions e
		SFX 22 y ications y
		SFX 22 0 ens [^ey]
PFX 5 Y 1	SFX 15 Y 3	SFX 23 Y 4
PFX 5 0 in .	SFX 15 e ion e	SFX 23 0 rs e
	SFX 15 y ication y	SFX 23 y iers [^aeiouy]
	SFX 15 0 en [^ey]	SFX 23 0 ers [aeiouy]
		SFX 23 0 ers [^ey]
PFX 6 Y 1	SFX 16 Y 3	
PFX 6 0 pro .	SFX 16 y iness [^aeiouy]	
	SFX 16 0 ness [aeiouy]	
PFX 7 Y 1	SFX 16 0 ness [^y]	
PFX 7 0 un .		
SFX 8 Y 3		
SFX 8 0 able [^aeiou]		
SFX 8 0 able ee		
SFX 8 e able [^aeiou]		

Figure 8: Noun classification dictionaries for English words

SFX-type rules describe how some noun modifications for English words with suffixes or endings (Fig. 8):

- *-able* [<sup>^</sup>aeiou], *-able ee*, *-able* [<sup>^</sup>aeiou]e (rule SFX ),
- *-d e*, *-ied* [<sup>^</sup>aeiou]y, *-ed* [<sup>^</sup>ey], *-ed* [aeiou]y (rule SFX 9),
- *-ing e*, *-ing* [<sup>^</sup>e] (rule SFX 10) and *-ieth y*, *-th* [<sup>^</sup>y] (rule SFX 11),
- *-ment* (rule SFX 14) and *-ion e*, *-ication y*, *-en* [<sup>^</sup>ey] (rule SFX 15),
- *-ings e*, *-ings* [<sup>^</sup>e] (rule SFX 12) and *-s* (rule SFX 13),
- *-iness* [<sup>^</sup>aeiou]y, *-ness* [aeiou]y, *-ness* [<sup>^</sup>y] (rule SFX 16),
- *-ies* [<sup>^</sup>aeiou]y, *-s* [aeiou]y, *-es* [sxzh], *-s* [<sup>^</sup>sxzhy] (rule SFX 17),
- *-r e*, *-ier* [<sup>^</sup>aeiou]y, *-er* [aeiou]y, *-er* [<sup>^</sup>ey] (rule SFX 18),
- *-st e*, *-iest* [<sup>^</sup>aeiou]y, *-est* [aeiou]y, *est* [<sup>^</sup>ey] (rule SFX 19),
- *-ive e*, *-ive* [<sup>^</sup>e] (rule SFX 20) and *-ly* (rule SFX 21),
- *-ions e*, *-ications y*, *-ens* [<sup>^</sup>ey] (rule SFX 22),
- *-rs e*, *-iers* [<sup>^</sup>aeiou]y, *-ers* [aeiou]y, *-ers* [<sup>^</sup>ey] (rule SFX 23). The letters e and y near the suffixes are decision markers.

A file of affixes (parts of words that attach to the root and bring grammatical or word-forming meaning, elements of word formation, for example, prefix, suffix, postfix, inflexion) has the \*.aff file type and may contain additional attributes - the rules of reduction to the base of the word (Fig. 9). The notation SET is usually used to identify the sequence of parts of affixes and directories. REP forms a lookup table to correct multiple characters for words. TRY identifies sequences to replace. SFX and PFX identify the types of suffixes and prefixes that are marked by word affixes.

	id	ordering	state	flag	type	lang	mask	find	repl	params	language
Редaguвати	26	26	1	a	SFX	uk	ih	ih	оном		*
Редaguвати	27	27	1	a	SFX	uk	ih	ih	оні		*
Редaguвати	28	28	1	a	SFX	uk	ir	ir	огу		*
Редaguвати	29	29	1	a	SFX	uk	ir	ir	огові		*
Редaguвати	30	30	1	a	SFX	uk	ir	ir	огом		*
Редaguвати	31	31	1	a	SFX	uk	ir	ir	озі		*
Редaguвати	32	32	1	a	SFX	uk	[ <sup>^</sup> h]id	id	оду		*
Редaguвати	33	33	1	a	SFX	uk	[ <sup>^</sup> h]id	id	одові		*
Редaguвати	34	34	1	a	SFX	uk	[ <sup>^</sup> h]id	id	одом		*
Редaguвати	35	35	1	a	SFX	uk	[ <sup>^</sup> h]id	id	оді		*
Редaguвати	36	36	1	a	SFX	uk	[ <sup>^</sup> h]id	id	ьоду		*
Редaguвати	37	37	1	a	SFX	uk	[ <sup>^</sup> h]id	id	ьодові		*
Редaguвати	38	38	1	a	SFX	uk	[ <sup>^</sup> h]id	id	ьодом		*
Редaguвати	39	39	1	a	SFX	uk	[ <sup>^</sup> h]id	id	ьоді		*
Редaguвати	40	40	1	a	SFX	uk	[h]id	id	оду		*
Редaguвати	41	41	1	a	SFX	uk	[h]id	id	одові		*
Редaguвати	42	42	1	a	SFX	uk	[h]id	id	одом		*
Редaguвати	43	43	1	a	SFX	uk	[h]id	id	оді		*
Редaguвати	44	44	1	a	SFX	uk	ib	ib	обу		*
Редaguвати	45	45	1	a	SFX	uk	ib	ib	обові		*
Редaguвати	46	46	1	a	SFX	uk	ib	ib	обом		*
Редaguвати	47	47	1	a	SFX	uk	ib	ib	обі		*

Figure 9: Rules for reduction to the base of a word of the noun type

The *flag* of the flag attribute determines the type of word, the *mask* of the mask attribute shows the ending identification rule, the value of the *find* attribute is the ending of the word in the nominative case, the value of the *repl* attribute is the ending of the word in the non-nominative case. Exceptions to the rules are given in square brackets. For example, the first line (ordering 26) describes a specific example of recognizing nouns of group a with the alternation of *-i -o* and the inflection *-ih* of the nominative case in the instrumental case

(inflection *-оном*), and the next entry (ordering 27) is the same nouns, but in the local case (inflection *-оні*), but does not recognize other rules of that group or other groups in the dative case - inflections *-онови* and *-ону* (Fig. 10). The third record (ordering 28) already recognizes nouns with alternation *-i -o* with inflections *v* of the nominative case in the dative case - inflection *-озу*, but does not recognize other rules of the same group and (rules 29-31 do this, respectively): *-озові* (Д.М.), *-озом* (О.), *-озі* (М.).

```

# тверда група в Називному відмінку однини з закінченням на -а
# Множина
SFX b a 0 [^клн]а # хата хат (P.)
# [^ВКЛНРШЧ] А > -А,- # хата > хат (P.)
# [ШЧ] А > -А,ЕЙ # миша > мишей (P.)
SFX b a 0 [^ст]ла # щогла щогл (P.)
SFX b ла ел [ст]ла # мітла мітел (P.)
# [^К] В А > -А,- # глава > глав (P.)
# [^Р] К В А > -А,- # буква > букв (P.)
# Р К В А > -ВА,ОВ # церква > церков (P.)
SFX b a 0 [^сжзм]на # батьківщина батьківщин (P.)
SFX b на ен [сжзм]на # сосна сосен (P.)
SFX b на н изна # тризна тризн (P.)
SFX b на ен озна # борозна борозен (P.)
SFX b а 0 [аееоуіію]ка # автоматика автоматик (P.)
SFX b ка ок [аееоуіію]ка # відпустка відпусток (P.)
# МР 2002.01.25
# Ш К А > -ШКА,ЩОК # дошка > дощок (P.)
# сестра - в /о

# м'яка група в Називному відмінку однини з закінченням на -я
# множина
SFX b я ь [іеіаоуіє]ня # Вася Вась (P.)
# МР 2001.09.02 родовий відмінок множини на -ря
SFX b я 0 [аео]ря # буря бур (P.)
# +++ ДК - зміни надані від vesna@ 11.06.02
SFX b я ь еря # вечеря вечерь (P.)
SFX b оря ір оря # зоря зір (P.)
# --- ДК - зміни надані від vesna@
#
# МР богиня кухня бойня вишня; 2001.09.02 -бНЯ,ЕНЬ
SFX b я ь [іеіаоуіє]ня # богиня богинь (P.)
SFX b ня онь [кх]ня # кухня кухонь (P.)
SFX b йня ень йня # бойня боень (P.)
SFX b ьня ень ьня # вітальня віталень (P.)
SFX b ня ень [іеіаоуіє]нкхйня # вишня вишень (P.)
# МР 2001.08.26 -лля -ття -ддя 2001.09.02 -[вмб]ля 2001.09.14 -стя
# МР 2001.09.16 [^С] [С] -сля (тесля - тесль) ?
SFX b ля ель [аееоуіє]скля # будівля будівель (P.)
SFX b я ь [аееоуіє]скля # Валя Валь (P.)
SFX b я ей адя # попада попадей (P.)
SFX b я ів [ада]дя # дядя дядів (P.)
SFX b я ь [ацс]тя # Катя Кать (P.)
SFX b ля ей лля # рілля рілей (P.)
# Т Л Я > -Я,ЕЙ # рілля > рілей (P.)
SFX b я ів уддя # суддя суддів (P.)
SFX b дя ей аддя # баддя бадей (P.)
SFX b тя ей ття # стаття статей (P.)
SFX b я ь [ао]стя # причастя причасть (P.)
SFX b я ей остя # гостя гостей (P.)
# МР
# Мар'я > Марей ?
SFX b 'я ей [аp]'я # сім'я сімей (P.)

```

**Figure 10:** An example of the rules of morphological analysis of Ukrainian nouns

The ninth entry (ordering 34) already recognizes nouns with inflexions on *-[а]ід* of the nominative case not after *-л* in the instrumental case with the inflexion *-одоm*, but does not recognize other rules of the same group and (according to rules 32-33 and 35 ): *-[а]оду*

(Д.Р.), -[<sup>^</sup>л]одові (Д.), -[<sup>^</sup>л]оді (М.). REP defines a substitution table for correcting several characters in Ukrainian words [535], for example REP 5; REP сч щ; REP уюч увальн; REP ююч ювальн; REP ємн містк; REP обез зне. Negative form prefix (Fig. 11):

- Adjectives ending in *-ий*;
- Adjectives of the short form change to *-ен* in the same way as the full form (ясен -ясний...).

The presence of a ratio of words blocked by the moderator (Fig. 12), in particular those that cannot be key, allows to reduce the amount of verification during text classification (Fig. 13). To identify keywords, it is important to correctly recognize adjectives in any case, gender and number (Fig. 14).

```

PFX Z Y 1
PFX Z 0      не      .      #   голосний > неголосний

# УВАГА!! не можна використовувати /Y з прапорцем /Z!!
# Сировиною групи /Y є прикметники у порівняльній формі (не 'простий/Y', а 'простіший/Y')

PFX Y Y 1
PFX Y 0      най     .      #   голосніший > найголосніший

# тотожні слова з префіксом "в-" утворені від слів з префіксом "у-"
# можуть створювати надто довгі рядки словоформ, поки що не використовується
PFX X Y 1
PFX X      в      у      в      #   вбити > убити
    
```

**Figure 11:** An example of rules for identifying the negative form of Ukrainian words

id	ordering	state	word	lang	params	language
1	1	1	після	uk	*	*
2	2	1	між	uk	*	*
3	3	1	are	en	*	*
4	4	1	and	en	*	*
5	5	7	між	uk	*	*
6	6	1	been	en	*	*
7	7	1	has	en	*	*
8	8	1	their	en	*	*
9	9	1	any	en	*	*
10	10	1	the	en	*	*
11	11	1	with	en	*	*
12	12	1	таких	uk	*	*
13	13	1	їхніми	uk	*	*
14	14	1	как	ru	*	*
15	15	1	такої	uk	*	*
16	16	1	на	uk	*	*
17	17	1	на	ru	*	*
18	18	1	ними	uk	*	*
20	19	1	для	uk	*	*
21	20	1	что	ru	*	*
22	21	1	или	ru	*	*
23	22	1	это	ru	*	*

**Figure 12:** The ratio of words blocked by the moderator

id	ordering	state	title	params	language
2	2	1	Наука	*	*
3	3	1	Технології	*	*
4	4	1	Інтернет	*	*
1	1	1	Без рубрики	*	*
5	5	1	Туризм	*	*

**Figure 13:** Relationship of rubrics

```

# Прикметники із закінченням на -ий
# прикметники короткої форми на -ен змінюються так само як і повної (ясен - ясний...)
#
# Чоловічого роду
SFX V   ий      ого      [^ц]ий      # відісланий відісланого      (Р.З.)
SFX V   ий      ому      [^ц]ий      # відісланий відісланому      (Д.М.)
SFX V   ий      им       ий          # відісланий відісланим      (О. Мн:Д.)
SFX V   ий      ім       ий          # відісланий відісланим      (М.)
# Жіночого роду
SFX V   ий      а       [^ц]ий      # відісланий відіслана      (Н.)
SFX V   ий      ої      [^ц]ий      # відісланий відісланої      (Р.)
SFX V   ий      ій      ий          # відісланий відісланій      (Д.)
SFX V   ий      у       [^ц]ий      # відісланий відіслану      (З.)
SFX V   ий      ою      [^ц]ий      # відісланий відісланою      (О.)
# Середнього роду
SFX V   ий      е       ий          # відісланий відіслане      (Н.)
# Множина
SFX V   ий      і       ий          # відісланий відіслані      (Н.)
SFX V   ий      их      ий          # відісланий відісланих      (Р.)
SFX V   ий      ими     ий          # відісланий відісланими      (О.)
#
# Прикметники на -лиций
#
# Чоловічого роду
SFX V   ий      ього     [^у]ций     # білолиций білолицього      (Р.З.)
SFX V   ий      ьому     [^у]ций     # білолиций білолицьому      (Д.М.)
# Жіночого роду
SFX V   ий      я       [^у]ций     # білолиций білолиця      (Н.)
SFX V   ий      ьої     [^у]ций     # білолиций білолицьої      (Р.)
SFX V   ий      ю       [^у]ций     # білолиций білолицю      (З.)
SFX V   ий      ьою     [^у]ций     # білолиций білолицьою      (О.)
#
# Чоловічого роду
SFX V   ий      ого      уций       # куций куцого      (Р.З.)
SFX V   ий      ому      уций       # куций куцо́му      (Д.М.)
# Жіночого роду
SFX V   ий      а       уций       # куций куца      (Н.)
SFX V   ий      ої      уций       # куций куцої      (Р.)
SFX V   ий      у       уций       # куций куцу      (З.)
SFX V   ий      ою      уций       # куций куцою      (О.)
#
# Прикметники із закінченням на -ій/-їй
#
# Чоловічого роду
SFX V   ий      ього     ій          # синій синього      (Р.)

```

**Figure 14:** An example of the rules of morphological analysis of Ukrainian adjectives

Let us describe each marked class of the set of MA noun rules:

- Class I for nouns marked with *flag* as *a, b, c, d* or *o*:
  - a. 1 declension: feminine, masculine and neuter nouns;
  - b. 2nd declension: masculine nouns ending in *-ap, -up*, stressed (mixed group in *-ap, -up*);
  - c. 2 declension: masculine nouns with alternation *-i, -o*;
  - d. Numerals *-ять, -сят, -сто*;
- Class II for nouns marked with *flag* as *e, f, g* or *h*:
  - a. Second declension masculine nouns with a zero ending;
  - b. Second declension masculine nouns ending in *-o*;
- Class III for nouns marked with *flag* as *i, j* or *k*:
  - a. Third declension without alternation;
  - b. Second declension neuter ending in *-o, -a, -я*;
  - c. Second declension on a consonant without ending *-i* in the local case;
- Class IV for nouns marked with *flag* as *l, m* or *n*:
  - a. Third declension with alternation;
  - b. The fourth declension of the neuter ending in *-а, -я*;

- c. Group V for nouns marked with the *flag* as p: masculine (m.) and feminine (f.) patronymics of the singular (s.) and plural (m.) of male names.

For further SYA, it is appropriate to recognize the verbs correctly (Fig. 15).

Let us describe in more detail each marked class of the set of noun recognition rules, indicating their total number N (Table 5). In total, about 1,300 rules for processing suffixes and endings are used for MA Ukrainian-language nouns, taking into account the alternation of letters.

```
SFX A Y 402

# зворотня форма - в групі /B
# складний майбутній час = /AG та складний зворотній час зворотньої форми = /BH
#
# MH (окрім Я, Ти, Він) для всіх закінчень, крім "йти" (йшов, йшла, йшло), та -сти
SFX A ти ла [^с]ти # абонувати абонувала (Вона)
SFX A ти ло [^с]ти # абонувати абонувало (Воно)
SFX A ти ли [^с]ти # абонувати абонували (Ми, Ви, Вони)
SFX A ти в [аеііоуя]ти # абонувати абонував (Я, Ти, Він)

# Т -вати (Т недоконаної форми, МБ доконаної)
SFX A вати ю [аюя]вати # абонувати абоную (Я)
SFX A вати еш [аюя]вати # абонувати абонуєш (Ти)
SFX A вати є [аюя]вати # абонувати абонує (Він)
SFX A вати емо [аюя]вати # абонувати абонуємо (Ми)
SFX A вати ете [аюя]вати # абонувати абонуєте (Ви)
SFX A вати ють [аюя]вати # абонувати абонують (Вони)
# MH -ати
# HФ -вати
SFX A вати й [ую]вати # абонувати абонуй (Ти)
SFX A вати ймо [ую]вати # абонувати абонуймо (Ми)
SFX A вати йте [ую]вати # абонувати абонуйте (Ви)
SFX A ти й [ая]вати # ставати ставай (Ти)
SFX A ти ймо [ая]вати # ставати ставаймо (Ми)
SFX A ти йте [ая]вати # ставати ставайте (Ви)
# Т -рвати, -звати (Т недоконаної форми, МБ доконаної)
SFX A ати у [рз]вати # рвати рву (Я)
SFX A ати еш [рз]вати # рвати рвеш (Ти)
SFX A ати е [рз]вати # рвати рве (Він)
SFX A ати емо [рз]вати # рвати рвемо (Ми)
SFX A ати ете [рз]вати # рвати рвете (Ви)
SFX A ати уть [рз]вати # рвати рвуть (Вони)
# HФ -вати
SFX A ати и [рз]вати # рвати рви (Ти)
SFX A ати імо [рз]вати # рвати рвімо (Ми)
SFX A ати іть [рз]вати # рвати рвіть (Ви)
#
# Дієслова з закінченням -зати з чергуванням з/ж (без чергування - гр. /I)
# Т -зати (Т недоконаної форми, МБ доконаної)
SFX A зати жу зати # казати кажу (Я)
SFX A зати жеш зати # казати кажеш (Ти)
SFX A зати же зати # казати каже (Він)
SFX A зати жемо зати # казати кажемо (Ми)
SFX A зати жете зати # казати кажете (Ви)
SFX A зати жуть зати # казати кажуть (Вони)
# MH -зати
# HФ -зати
# варіанти закінчень: -жи (д.ф. зв'язи), -ж (різати, зарізати - ріж), і в гр. /I -зай (вирізати - виріжай)
SFX A зати ж ізати # різати ріж (Ти)
SFX A зати ж мазати # мазати маж (Ти)
SFX A зати жи казати # казати кажи (Ти)
SFX A зати жи [еия]зати # лизати лижи (Ти)
```

Figure 15: An example of the rules of morphological analysis of Ukrainian verbs

Table 5

Basic MA rules for marking nouns when marking a part of speech

Class	flag	N	Features of MA-rules
I	a	248	For the singular: <ul style="list-style-type: none"> <li>1 declension: feminine, masculine and neuter nouns.</li> <li>2 declensions: masculine in <i>-ap</i>, <i>-up</i>, stressed (mixed group in <i>-ap</i>, <i>-up</i>).</li> </ul>



Class	flag	N	Features of MA-rules
I	b	384	<ul style="list-style-type: none"> <li>• 2 declensions: masculine nouns with alternating <i>-i</i> and <i>-o</i>.</li> <li>• numerals <i>-ять, -сят, -сто</i>.</li> </ul> <p>For the plural:</p> <ul style="list-style-type: none"> <li>• 1 declension: feminine, masculine and neuter nouns.</li> <li>• 2 declensions: masculine in <i>-ар, -ур</i>, stressed (mixed group in <i>-ар, -ур</i>).</li> <li>• 2 declensions: masculine nouns with alternating <i>-i</i> and <i>-o</i>.</li> <li>• plural nouns ending in <i>на -и</i>.</li> </ul>
I	c	54	<p>2 declension, in gen. singular case with the ending <i>-а/-я</i>, namely the meaning:</p> <ul style="list-style-type: none"> <li>• beings and persons: <i>студента, моря, Любомира</i>;</li> <li>• items that can be counted: <i>зошита, ножа, олівця</i>;</li> <li>• own settlements: <i>Ужгорода, Тернополя</i>;</li> <li>• water bodies with a pronounced inflexion: <i>Дніпра</i>;</li> <li>• measurements: <i>квадрата, міліметра</i> (but <i>віку, року</i>);</li> <li>• definitions: <i>відмінка</i>;</li> <li>• architecture: <i>парника, коридора, гаража</i>.</li> </ul>
I	d	44	<ul style="list-style-type: none"> <li>• vocative case;</li> <li>• first declension (for endings <i>[ая]</i>);</li> <li>• 2 declensions (ending <i>[рнгдблвк]</i> with alternation <i>o-i</i> and dropping <i>e, o</i>).</li> </ul>
I	o	53	<p>For the plural:</p> <ul style="list-style-type: none"> <li>• 1 declension: female/male/neuter genus with alternation of <i>o/i</i> and the appearance of <i>o(e)</i> in the genus;</li> <li>• 2 declensions: neuter in <i>-o</i> with alternating <i>o/i</i> in gen. plural.</li> </ul>
II	e	19	<p>For the singular:</p> <ul style="list-style-type: none"> <li>• a solid group of nouns ending in <i>-o</i>;</li> <li>• a solid group of nouns with a zero ending;</li> <li>• a solid group of nouns with zero ending in sibilant;</li> <li>• mixed group with zero ending in sibilants;</li> <li>• a soft group ending in <i>-й, -ій</i> or <i>-ь</i>.</li> </ul>
II	f	25	<p>For the plural:</p> <ul style="list-style-type: none"> <li>• a solid group of nouns with a zero ending;</li> <li>• a solid group of nouns with zero ending in sibilant;</li> <li>• mixed group with zero ending in sibilants;</li> <li>• a solid group of nouns ending in <i>-o</i>;</li> <li>• a soft group ending in <i>-й, -ій</i> or <i>-ь</i>;</li> <li>• a group of nouns ending in <i>-ття, -ттів</i>, incl. From group <i>/i</i>;</li> <li>• coincides with the gen. singular case;</li> <li>• nouns ending in <i>-ок</i> and dropping <i>o</i> are transferred to group a.</li> </ul>
II	g	3	<p>genitive case of the second declension in <i>-а</i>.</p>
II	h	5	<ul style="list-style-type: none"> <li>• second declension (required for endings in consonants except <i>ЙЖЧШЩ</i>);</li> <li>• nouns of the second declension of the masculine gender with a zero ending;</li> <li>• coincides with the dative.</li> </ul>
III	i	47	<p>For the singular:</p> <ul style="list-style-type: none"> <li>• nouns of the third declension of the feminine gender with a zero ending;</li> <li>• ending in a sibilant, except <i>-ь</i>;</li> <li>• nouns of the second declension of the neuter ending in <i>-о, -а</i> or <i>-я</i>;</li> <li>• soft group in <i>-е</i>, except for sibilants;</li> <li>• mixed group on sibilant before <i>-е</i>;</li> <li>• from adjectival nouns.</li> </ul>
III	j	66	<p>For the plural:</p> <ul style="list-style-type: none"> <li>• nouns of the third declension of the feminine gender with a zero ending;</li> <li>• ending in a sibilant, except <i>-ь</i>;</li> <li>• nouns of the second declension of the neuter ending in <i>-о, -а</i> or <i>-я</i>;</li> <li>• a soft group on <i>-е</i>, except for sibilants or a mixed group;</li> </ul>



Class	flag	N	Features of MA-rules
III	<i>k</i>	8	<ul style="list-style-type: none"> <li>• mixed group on sibilant before <i>-e</i>.</li> <li>• vocative;</li> <li>• nouns of the third declension of the feminine gender with a zero ending;</li> <li>• feminine singular of adjectival nouns.</li> </ul>
IV	<i>l</i>	40	For the singular: <ul style="list-style-type: none"> <li>• nouns of the third declension with alternation;</li> <li>• nouns of the fourth declension of the middle gender ending in <i>-a</i> or <i>-я</i>;</li> <li>• 2 masculine declensions in <i>-o[дв]ець</i> with dropout of <i>e</i> and alternation of <i>o-i</i>;</li> <li>• mixed group 2 declensions in <i>-яp</i>;</li> <li>• 2 masculine declensions in <i>-ap/-up</i>, stressed (mixed group in <i>-ap/-up</i>).</li> </ul>
IV	<i>m</i>	66	For the plural: <ul style="list-style-type: none"> <li>• nouns of the third declension with alternation;</li> <li>• nouns of the fourth declension of the middle gender ending in <i>-a/-я</i>;</li> <li>• 2 masculine declensions in <i>-o[дв]ець</i> with dropout of <i>e</i> and alternation of <i>o-i</i>;</li> <li>• mixed group 2 declensions in <i>-яp</i>;</li> <li>• 2 masculine declensions in <i>-ap/-up</i>, stressed (mixed group in <i>-ap/-up</i>).</li> </ul>
IV	<i>n</i>	9	<ul style="list-style-type: none"> <li>• with alternation <i>i e</i> or with alternation <i>i o</i>;</li> <li>• ending in a sibilant, except <i>-ь</i>;</li> <li>• nouns of the third declension of the feminine gender with a zero ending;</li> <li>• mixed group of 2 declensions in <i>-яp</i> soft group in <i>-ap/-up</i>.</li> </ul>
IV	<i>q</i>	2	<ul style="list-style-type: none"> <li>• mixed group 2 declensions in <i>-яp</i>;</li> <li>• soft group in <i>-ap/-up</i> (accented endings in declension).</li> </ul>
V	<i>p</i>	222	n masculine/feminine singular and plural patronymics from male names.

It is quite difficult to generate terminal chains in English (but MA rules are much less, not in Ukrainian), because the presence of articles and the connection of groups of nouns with each other with the corresponding preposition makes the tree longer and wider. The generation of terminal chains in the Ukrainian language is complicated by cases and generic differences in inflexions of the term used in the context. To identify keywords, it is not enough to recognize nouns (about 1300 RE-rules), it is also necessary to identify adjectives - a total of 99 RE-rules for Ukrainian texts (Table 4.6-Table 4.7). For correct SYA and SEM, including ontology construction, it is necessary to recognize verbs based on more than 800 RE rules.

**Table 6**

Basic MA rules for marking adjectives as parts of speech

flag	N	Peculiarities of MA rules for recognizing adjectives
<i>V</i>	83	<ul style="list-style-type: none"> <li>• singular ending in <i>-uй</i>;</li> <li>• the short form singular changes to <i>-ен</i> in the same way as the full form (<i>ясен - ясний...</i>);</li> <li>• ending in <i>-лиций</i>;</li> <li>• ending in <i>-ій/-їй</i>;</li> <li>• plurals ending in <i>-ій/-їй</i>;</li> <li>• possessives from nouns of the 1st declension - names of people in <i>-ин</i>;</li> <li>• possessives from nouns of the 2nd declension in <i>-ів</i> (solid group);</li> <li>• possessives from nouns of the 2nd declension in <i>-їв</i>.</li> </ul>
<i>U</i>	13	<ul style="list-style-type: none"> <li>• soft group of possessives ending in <i>-ів</i> -&gt; <i>-ев</i>;</li> <li>• plurals ending in <i>-ів</i>.</li> </ul>
<i>W</i>	3	the formation of an adverb from an adjective, the neuter gender of the comparative form of adjectives corresponds to the corresponding adverb in the comparative form ( <i>міцніший - міцніше</i> ).

**Table 7**

Basic SFX-type RE of Ukrainian adjectives based on goroh.pp.ua

N	Flag	Genus	F1	F2	RE	Numeric	Sign	Example 1	Example 2	Case	N
1	V	ч	ий	ого	[^ц]ий	одн	in -ий	текстовий	текстового	Р.З.	1
2				ому					текстовому	Д.М.	2
3				им	ий				текстовим	О.Мн.:Д.	3
4				ім					текстовім	М.	4
5		ж		а	[^ц]ий				текстова	Н.	5
6				ої					текстової	Р.	6
7				ій	ий				текстовій	Д.	7
8				у	[^ц]ий				текстову	З.	8
9				ою					текстовою	О.	9
10		с		е	ий				текстове	Н.	10
11		-		і		мн			текстові		11
12				их					текстових	Р.	12
13				ими					текстовими	О.	13
14		ч		ього	[^у]ций	одн	in -лиций	білолиций	білолицього	Р.З.	14
15				ьому					білолицьому	Д.М.	15
16		ж		я					білолиця	Н.	16
17				ьої					білолицьої	Р.	17
18				ю					білолицю	З.	18
19				ьою					білолицьою	О.	19
20		ч		ого	уций			куций	куцого	Р.З.	20
21				ому					куцому	Д.М.	21
22		ж		а					куца	Н.	22
23				ої					куцої	Р.	23
24				у					куцу	З.	24
25				ою					куцою	О.	25
26		ч	ій	ього	ій		in -ій/-їй	крайній	крайнього	Р.	26
27				ьому					крайньому	Д.	27
28				ім					крайнім	О.Мн.:Д.	28
29		ж		я					крайня	Н.	29
30				ьої					крайньої	Р.	30
31				ю					крайню	Д.	31
32				ьою					крайньою	О.	32
33		с		є					крайне	Н.	33
34		-	й	-	[ї]й	мн			крайні	Н.	34
35				х					крайніх	Р.	35
36				ми					крайніми	О.	36
37		ч	ій	його	їй	одн		безкрай	безкрайого	Р.З.	37
38				йому					безкрайому	Д.	38
39				ім					безкраім	О.М.Мн.:Д.	39
40		ж		я					безкрая	Н.	40
41				йої					безкрайої	Р.	41
42				ю					безкраю	З.	42
43				йою					безкрайою	О.	43
44		с		є					безкрає	Н.	44
45		ч	-	ого	[ї]н		possessives from nouns of the 1st declension - names of people on -ин	мамин	мамино	Р.	45
46				ому					маминому	Д.	46
47				им					маминим	О. Мн.:Д.	47
48				ім					маминім	М.	48
49		ж		а					мамина	Н.	49
50				ої					маминої	Р.	50
51				ій					маминій	Д.М.	51
52				у					мамину	З.	52
53				ою					маминою	О.	53
54		с		е					мамине	Н.	54
55				і		мн			маміні	Н.	55
56				их					маминих	Р.	56
57				ими					маминими	О.	57
58		ч	ів	ового	ів	одн	possessives from nouns of the 2nd	татів	татового	Р.	58
59				овому					татовому	Д.	59

N	Flag	Genus	F1	F2	RE	Numeric	Sign	Example 1	Example 2	Case	N
60				овим			declension in -ів, solid group		татовим	О. Мн:Д.	60
61				овім					татовім	М.	61
62		ж		ова					татова	Н.	62
63				ової					татової	Р.	63
64				овій					татовій	Д.М.	64
65				ову					татову	З.	65
66				овою					татовою	О.	66
67		с		ове					татове	Н.	67
68		-		ові		мн			татові	Н.	68
69				ових					татових	Р.	69
70				овими					татовими	О.	70
71		ч	ів	євого	ів	одн	possessives from nouns of the 2nd declension in -ів, hard group	Вереміїв	Веремієвого	Р.	71
72				євому					Веремієвому	Д.	72
73				євим					Веремієвим	О. Мн:Д.	73
74				євім					Веремієвім	М.	74
75		ж		єва					Веремієва	Н.	75
76				євої					Веремієвої	Р.	76
77				євій					Веремієвій	Д.М.	77
78				єву					Веремієву	З.	78
79				євою					Веремієвою	О.	79
80		с		єве					Веремієве	Н.	80
81		-		єві		мн			Веремієві	Н.	81
82				євих					Веремієвих	Р.	82
83				євими					Веремієвими	О.	83
1	U	ч	ів	євого	ів	одн	soft group of possessives on - ів, -ев	вчителів	вчителевого	Р.	84
2				євому					вчителевому	Д.	85
3				євим					вчителевим	О. Мн:Д.	86
4				євім					вчителевім	М.	87
5		ж		єва					вчителева	Н.	88
6				євої					вчителевої	Р.	89
7				євій					вчителевій	Д.М.	90
8				єву					вчителеву	З.	91
9				євою					вчителевою	О.	92
10		с		єве					вчителеве	Н.	93
11		-		єві		мн			вчителеві	Н.	94
12				євих					вчителевих	Р.	95
13				євими					вчителевими	О.	96
1	W		ий	о	[^жчшщ]ий	-	adverb	надісланий	надіслано	-	97
2			ій	ьо	ій			синій	синьо		98
3			ій	йо	ій			безкрай	безкрайо		99

CLS marks the words of the input text as parts of speech (clarifies after GA the tagged/marked lexemes as words) based on RE-rules and analysis of inflexions as singular nouns of the corresponding gender and case, plural nouns of the corresponding case, adjectives, adverbs, verbs, personal pronouns, etc. (each with a collection of features).

The MA module returns a collection of paragraph lists, each of which is a list of sentences, which are lists of tokens, including words marked by parts of speech. Periodic interim analysis of the input/integrated textual content allows to assess how the thematic corpus changes over time. In the process of analysis, we will count the number of paragraphs, sentences and words, and also save each unique lexeme in an additional intermediate dictionary. If the lexeme/word did not exist in the dictionary of lexemes/word bases, we mark it as new and store it in the intermediate dictionary for analysis by the moderator. We count the number of content and categories in the corpus of incoming text content and form a dictionary with a statistical summary of the corpus, which contains: the total number of integrated content and categories; the total number of paragraphs, sentences and words; the number of unique tokens; lexical diversity as the ratio of the number of unique lexemes

to their total number; the average number of paragraphs in the content; average number of sentences per paragraph; total processing time.

Since the corpus grows as new data is collected, pre-processed and compressed, the MA method will allow us to calculate these features and analyze their dynamics of change. It is an important content monitoring tool to identify possible problems in CLS, for example, in an ML model, a significant change in lexical diversity and the number of paragraphs per content affects the quality of the model. That is, the MA method and GA methods, in addition to the identification of tokens and direct marking of words by parts of speech, are used to collect additional information when determining the amount of changes in the corpus to timely start further vectorization and restructuring of the ML model. The main stage of the MA method is the identification of the bases of words (stemming) without taking into account inflexions (suffixes and endings) and in some cases - prefixes. According to the content of the inflexions, a part of the language is identified as a word (Fig. 16).

```
var $ADJECTIVE = //Прикметник
'(ими|ій|ий|а|е|ова|ове|ів|є|їй|єє|єс|я|ім|ем|им|ім|их|іх|ою|йми|іми|у|ю|ого|ому|ої)$';
// Дієприкметник
var $PARTICIPLE = '(ий|ого|ому|им|ім|а|ій|у|ою|їй|і|их|йми|их)$';
// Дієслово
var $VERB = '(сь|ся|ив|ать|ять|у|ю|ав|али|учи|ячи|вши|ши|є|ме|ати|яти|є)$';
var $NOUN = //Іменник
'(а|єв|ов|є|ями|ами|єи|и|ей|ой|ий|й|иям|ям|ием|ем|ам|ом|о|у|ах|иях|ях|ь|ь|ію|ью|ю|
ія|ья|я|і|ові|ї|єю|єю|ою|є|єві|єм|єм|ів|ів|\`ю)$';
```

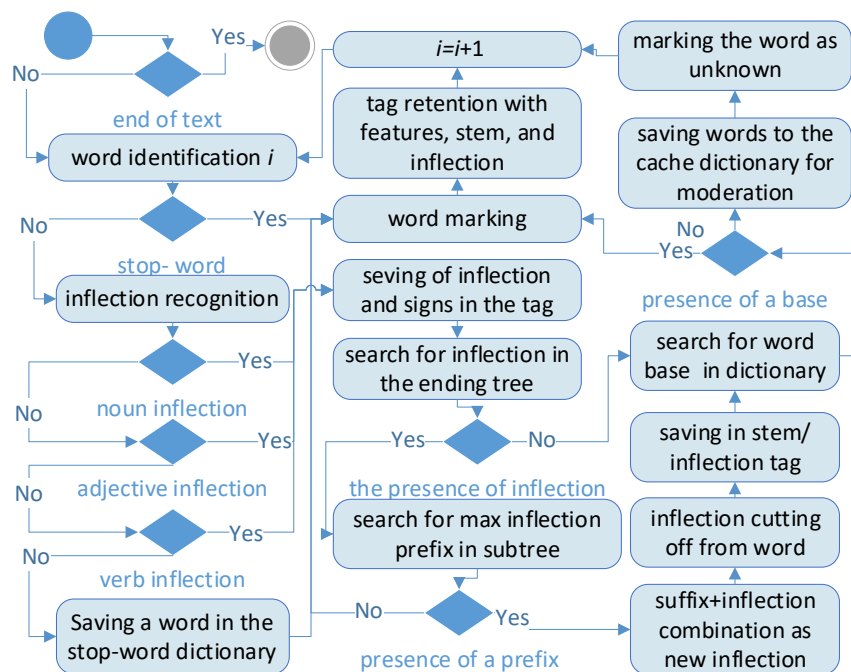
**Figure 16:** An example of identification of forms of inflexion according to part of speech

For the next SYA, this is not enough (to mark the word only as a part of speech), it is still necessary to determine, for example, gender/distinctiveness, etc., for a noun/adjective. The classic Porter stemmer algorithm works by sequentially cutting off endings and suffixes. For English-language texts, this is not a problem, as there are very few inflexions. For Ukrainian words, a modified (extended) algorithm of Porter's stemmer should be applied with a check of both additional inflexions depending on the part of the language (according to the tree of endings), as well as the obtained word bases with a dictionary of bases to identify the existing word (Fig. 17).

**Algorithm 4.1. Modified Porter stemmer algorithm**

- Stage 1.** Identify the next token as the word  $w_i$  ( $w_s = w_i$ ).
- Stage 2.** Check with the dictionary of stop words whether  $D_{w_{sw}}$  or  $w_s$  is a service word. If yes, then  $i = i + 1$  and go to step 1, otherwise go to step 3.
- Stage 3.** Go to the end of the word  $w_s$ . Recognize the inflection  $f_1^i$  in  $w_s$  from all possible ones (Fig. 4.16 - the longest one is chosen, for example, in  $w_s = \text{мекстова}$  we choose the ending  $f_1^i = \text{ова}$ , not  $f_1^i = \text{а}$ ) from the RE word type as  $R_{adjectival}$ ,  $R_{noun}$  or  $R_{verb}$  and in the presence of the removal of the inflection  $f_1^i$  (Fig. 18).
- Stage 4.** Preservation of inflection  $f_1^i$  in the word tag  $w_i$ .
- Stage 5.** Mark  $w_i$  as type  $m_{adjectival}^{w_i}$ ,  $m_{noun}^{w_i}$  or  $m_{verb}^{w_i}$  respectively.

- Stage 6.** Finding the deleted inflection  $f_1^i$  in the tree of inflexions  $T_{flexion}$  (the longest one is chosen). Checking the contents of the subtree  $T_{flexion}^{f_1^i}$  with the existing word ending  $f_2^i$  ( $f = f_2^i + f_1^i$ ). If  $w_s$  ends in  $f_2^i$  and has a counterpart in  $T_{flexion}^{f_1^i}$ , then we store it in  $f_i = f$  and delete in  $w_s$ .
- Stage 7.** We check the obtained base  $w_s$  of the initial word  $w_i$  with the content of the base dictionary  $D_{w_s}$  of Ukrainian words. If there is no respondent, we save  $\langle w_i, w_s \rangle$  in the additional temporary intermediate dictionary  $D_{\langle w_i, w_s \rangle}$  for the moderator and proceed to stage 1, otherwise proceed to stage 4.
- Stage 8.** Analysis of inflexion and the presence/absence of alternation of letters in the base/inflexions of the words  $\langle w_i, w_s \rangle$  and the analogue of the base of the word in  $D_{w_s}$  according to the relevant MA RE-rule to identify additional features of the analyzed word  $w_i$ .
- Stage 9.** Addition of identified linguistic features of the recognized part of speech to the tag of the word  $w_i$  of type  $m_{adjectival}^{w_i}$ ,  $m_{noun}^{w_i}$  or  $m_{verb}^{w_i}$  respectively. Saving the results in the corresponding dictionary  $D_{w_i}$  of the analyzed text.



**Figure 17:** Modified stemming algorithm

The increase in volume of MA RE-rules increases in a geometric progression the load on CLS only due to the recognition of inflexions and the bases of word forms. For English-language texts, the complexity is less due to several parameters, for example, for nouns 2 cases – 2 inflexions in the plural (s|es). For the German language, the complexity increases - 4 cases (but inflexions almost do not change, only articles change), phrases with  $\geq 2$  words are written together, etc. In the Ukrainian language, there are 7 cases of nouns, each of which changes its inflexion depending on the gender and plural/singular, and some words have different endings in some cases (for example, for *втручання* [vtruchannya] (intervention) in the local case, there are two options – *втручання*, *втручання*), in addition, there is often alternation of letters.

```

var $VOWELS = '(a|e|i|i|o|y|и|е|ю|я)/' # українські голосні літери
var RV # частина слова після першої VOWELS. RV=0, якщо VOWELS=0 в Wi відсутні.
# Всі перевірки f проводяться над RV. Літери перед RV не беруть участь взагалі.
# Так, при перевірці PARTICIPLE наступні а/я також повинні бути всередині RV.
var R1 # частина Wi після 1-го рядка VOWELS+NOVOWELS.
var R2 # частина R1 після 1-го рядка VOWELS+NOVOWELS.
# Наприклад, Wi=інформаційний: RV = нформаційний, R1 = формаційний, R2 = маційний.
Class 1. ADVERB # дієприслівник
  Group 1: '[a|я]?(в|вши|вшися)/'
  Group 2: '(ив|ивши|ившися)/'
Class 2. ADJECTIVE # прикметник
  '(a|e|i|i|и|ими|іми|ій|ий|ім|ім|им|ього|ого|ьому|ому|іх|их|ую|юю|ая|яя|ою|ею)/'
Class 3. PARTICIPLE # дієприкметник
  Group 1: '[a|я]?(вш|юва|ува|уч|юч|л)/'
  Group 2: '(нн|н|ячи|ачи|ова|ову|єм)/'
Class 4. REFLEXIVE = '(ся|сь)/' # рефлексивна флексія
Class 5. VERB # дієслово
  Group 1: '[a|я]?(ла|е|ете|йте|ли|люю|й|в|ем|емо|ний|ло|ть|но|ють|ні|ть|еш)/'
  Group 2: '(ила|ела|ена|йте|ите|ете|юй|уй|ій|ай|ало|ив|или|имо|ений|ило|іло|
    ено|ють|ать|ені|ять|іть|ить|иш|ую|ю)/' #
  Class 6. NOUN = '(a|e|v|ov|i|t|я|e|ами|іями|ями|еї|ею|ями|ям|іі|и|ою|ій|ой|ий|
    й|им|им|ім|ам|ом|о|у|ах|ях|ую|ю|ія|я)/' # іменник
  Class 7. SUPERLATIVE = '(ш|ш)/' # ступінь порівняння - найдовший, миліший
  Class 8. DERIVATIONAL = '[ість]?/' # словотворча флексія - милість, щедрість
  Class 9. ADJECTIVAL # (ADJECTIVE|PARTICIPLE+ADJECTIVE): падаюча = пада+юч+а.

```

**Figure 18:** Classes of linguistic features of inflexions of morphological analysis

Therefore, for Ukrainian words, Porter's simple classic stemming algorithm is not suitable (reducing the word to the base root by cutting off inflexions). It is better to combine such an algorithm with a search/check of the obtained intermediate results with a tree of inflexions (so as not to go through all possible inflexions) and with the content of thematic dictionaries of bases with a set of RE-rules for the identification of features (classification by parts of speech). Only for text rubrication based on word identification, it is enough to conduct MA only for some noun groups (adjectives with nouns and nouns with nouns) without analyzing words of other parts of speech (recognition by the tree of inflexions - not an adjective and not a noun - ignore, in addition, the key ones should be sometimes there can be 1 preposition next to and only between nouns. It is enough to identify the bases of nouns/adjectives/abbreviations in the text and analyze their probability of clustering in different parts of the content relative to the total volume.

The classic stemming algorithm - Porter's Stemmer - does not use dictionaries of word bases but only applies a set of RE-rules for cutting off inflexions in sequence according to the specifics of a specific language. The algorithm works with individual words without analyzing and taking into account the context. Linguistic features such as features of word formation (prefix, suffix, etc.) and parts of speech (noun, verb, etc.) are not taken into account. The basis is the following techniques for words:

- cutting off the inflexion from the analyzed word (for Ukrainian words, it can be implemented with the obtained bases and inflexions check with analogues in DB).
- the word has an invariable inflexion (the condition is impossible for most Ukrainian words, but it is possible to identify particles, conjunctions, prepositions, some nouns of foreign origin, abbreviations, etc.).

- changes inflexion in declension due to dropping/alternating letters.
- the change of word inflexion and word formation corresponds to a specific RE-rule, for example, when forming words from some verb groups:

(ов)\*ува(ти|нню|нням|нні|ння|ли|ло|ла|вшись|вши|в|вся|всь|лися|лись|тися|тись)  
 [(ov)\*uva(ty|nnyu|nnyam|nni|nnya|ly|lo|la|vshys'|vshy|v|vsya|vs'|lysa|lys'|tysya|tys')].

- changing the inflexion of the word as an exception to the RE rules.
- the ending of the word coincides with the envelope RE-rule of identification of inflexion, but the word itself has no inflexion: *вітер* [viter] (wind), but *відеп* [vider] (bucket).
- most short words are invariable (stop word dictionary is sufficient).

Such techniques significantly complicate the stemming algorithm of Ukrainian words. Therefore, first, widespread inflections are analyzed, for example, for 1 letter ц (34), щ (110), ф (214), б (281), п (341), ж (353), з (581), г (636), л (754), с (914), ч (959), д (1038), н (2531), р (2709) or 1-4 letters (Table 2.2). Inflexions  $\geq 5$  (for example,  $\max(\text{йтесь})=6837$ ,  $\max(\text{ванням})=4656$ ) are significantly less among keywords, therefore, for the speed/efficiency of the solution in some CLS NLP tasks, they are ignored, but for SYA/ SEM will not allow this. Many NLP tasks do not require full implementation of all NLP processes from grapheme to pragmatic analyses. For example, to identify keywords, it is enough to provide a grapheme and morphological analysis (algorithm 4.2). But before almost any NLP process, the text must be normalized.

#### **Algorithm 4.2. Abbreviated naive processing of textual content**

**Stage 1.** Rough tokenization (or grapheme analysis) of special characters of the input text.

*Step 1.1.* Reading the text and removing repeated consecutive spaces and tags if they are present (if the text is integrated from a Web resource), sequentially marking the service characters of the beginning/end of the paragraph/heading/text, etc.

*Step 1.2.* Grapheme parsing and segmentation between service characters or tags of the input text  $X$ , sequentially marking each sequence of non-alphabetic characters as tokens and recognizing alphabetic sequences between spaces and other special characters (eg numbers and punctuation) according to RE rules as token words to form a list  $S$  of identified alphabetic tokens as words  $w_i$ .

*Step 1.3.* Sort the list  $S \rightarrow S_A$  identified tokens  $w_i$  alphabetically, counting occurrences of identical chains and forming an alphabetic-frequency dictionary  $D_a$ , the record of which is in the form of the number of occurrences – a word.

*Step 1.4.* Transferring all letters of the upper register to the lower register and recalculating occurrences of word-tokens in the alphabetic-frequency dictionary  $D_A \rightarrow D_a$ .

*Step 1.5.* Sort and save the dictionary  $D_a \rightarrow D_N$  of identified  $w_i$  words by decreasing the frequency of appearance (in Germanic languages, the top will be articles, pronouns, adjectives and conjunctions, and in Slavic languages, most words with the same base and different inflexions will occupy different lines of the list, which significantly distorts the picture of the real distribution of words in texts).

**Stage 2.** Segmentation/tokenization of words of the analyzed text content.

*Step 2.1.* Word segmentation based on dictionaries, metrics such as the probability of an error in a word, and statistical sequence models pre-trained from segmented text corpora (between spaces, punctuation, etc.).

- Step 2.2.* Tokenization based on RE-rules of marked tokens of the sequence type of non-alphabetic characters as tokens (dates, prices, URLs, hashtags, e-mail addresses, etc.), punctuation (as the end of a sentence or the boundary of a subordinate clause), mixed tokens of alphabetic-non-alphabetic characters (abbreviations, complex hyphenated words, with an apostrophe, etc.), lines with uppercase characters (such as the beginning of a sentence, geographical names, proper names, abbreviations) and their normalization if necessary (for example, *к.т.н.* → *ктн* (PhD) as a separate word- token or *ML як машинне навчання* [mashynne navchannya] (machine learning)).
- Step 2.3.* Analysis of tokens with uppercase characters (except when only the first letters are capitalized) for labelling based on the RE-rules of finite automata or as an abbreviation or emotion transfer.
- Step 2.4.* Marking of unidentified  $D_x$  tokens and ambiguities (e.g. apostrophe as part of a word, etc.).
- Stage 3.** Lemmatization of a set of recognized and labelled alphabetic tokens of the text as lemmas, identified as words of the analyzed text.
- Step 3.1.* Normalization of tokens based on the identification of affixes from the termination tree as stenocardia of marked token-words (reducing the word to its initial form based on RE-rules MA for identification roots and affixes through Algorithm 1 of Porter's modified stemmer), i.e. determination of whether the analyzed tokens have the same root and differ only in inflexion with sequential identification of the part of the language of the analyzed words with subsequent marking of them as lemmas with all accompanying linguistic features.
- Step 3.2.* Regrouping and recalculation of word frequencies in the alphabetic-frequency dictionary  $D_N \rightarrow D_l$  taking into account the normalized words in step 3.1.
- Stage 4.** Additional analysis of unidentified tokens  $D_x \neq \emptyset$  by iteratively combining frequent character/string pairs within token words (for example, whether tokens between spaces or other punctuation marks *контент-аналіз* [kontent-analiz] (content-analysis), *Web-сайт* [Web-sayt] (Web-site), *контент-моніторинг* [kontent-monitorynh] (content-monitoring) or *Web-resource* [Web-resource] (Web-resource) are one word, or two) through bit-pair encoding, or BPE based on text compression for further possible identification of words, their labelling and normalization.
- Step 4.1.* Formation of a set of symbols equal to the collection of properties with  $D_x \neq \emptyset$ . K We present each word as a sequence of characters plus a special character at the end of the word or a special character, such as a dash, within a token (for example, *контент-*, *Web-*, *контент-* or *Web-*). We denote  $i = 0$ .
- Step 4.2.* Calculation of the number  $n_l$  of each pair of characters/lines  $\forall (s_k^x, s_j^x)$  as occurrences of word stems in the input text when  $\forall \{s_k^x \in D_x, s_j^x \in D_l\}$  or  $\forall \{s_k^x \in D_l, s_j^x \in D_x\}$ , which are next to each other and separated by a special character dash (compound words), period (date), comma (real number) and/or space, or their combination, but not punctuation marks, numbers and other special characters.
- Step 4.3.* Formation of the alphabetic-frequency dictionary  $D'_x$  based on  $\forall (s_k^x, s_j^x)$ . Determination of the number of occurrences of unique lexemes in  $D'_x \rightarrow h = |D'_x|$ .
- Step 4.4.* Finding  $n_l = \max$  of the most frequent pair  $a_i = (s_k^x, s_j^x)$  in  $D'_x$ , where  $\exists (s_k^x, s_j^x) \in D'_x$ ,  $\exists \{s_k^x \in D_x, s_j^x \in D_l\}$  або  $\exists \{s_k^x \in D_l, s_j^x \in D_x\}$ .
- Step 4.5.* Replacing  $a_i$  with a new combination/merge character/string  $b_i = s_k^x s_j^x$ .
- Step 4.6.* Extracting from  $D'_x$  the value  $s_k^x s_j^x$  and from  $D_x$  the values  $s_k^x$  or  $s_j^x$  respectively.
- Step 4.7.* Calculation of the number of occurrences in the input text  $b_i$ , occurrences of  $s_k^x$  and  $s_j^x$  at  $\exists s_k^x \in D_l$  and/or  $\exists s_j^x \in D_l$  respectively, when they are used separately (not next to each other).



Step 4.8. Inclusion in  $D_l$  of the value of  $b_i$ , and its frequency of occurrence. Overwriting frequency values in  $D_l$  for  $s_k^x$  and  $s_j^x$  at  $s_k^x \in D_l$  and/or  $s_j^x \in D_l$  respectively.

Step 4.9. We denote  $i = i + 1$ . If  $h > 0$ ,  $D_x \neq \emptyset$  for  $\forall n_l > 1$  and in  $D'_x$  is at least 1 marked  $b_i$ , then go to step 4.4, otherwise ( $h = 0$  and  $D_x = \emptyset$  or  $D_x \neq \emptyset$  at  $\forall n_l = 1$  or  $\forall s_k^x$  no non-unique pair  $\forall s_j^x$  for formation  $a_i = (s_k^x, s_j^x)$ ) – until step 5.

**Stage 5.** Segmentation of sentences in the analysed content.

### 4.3. Method of lexical analysis of the Ukrainian language

The process of lexical analysis of the Ukrainian-language text  $C'_\gamma$  consists in parsing, segmentation and tokenization of each sentence separately, which is characterized not by a strict order of words, but at the same time by a constant arrangement of individual linguistic units. In a complete simple Ukrainian sentence with direct word order, the structural scheme is conditionally fixed. The main lexical categories of the corresponding sentence are noun and verb groups. Type 0 grammar according to N. Chomsky's classification is not appropriate for such sentences due to the complexity of implementation. With context-dependent grammar, specific restrictions are applied, in particular, to the structure of a Ukrainian-language sentence with some set of variations. Based on the syntactic rules of generating Ukrainian-language sentences with partial word order (for example, there is no strict order for the subject and predicate in the sentence, but the adjective is usually before the noun or another adjective, if it is not a poetic passage, also the lexical units of the *noun group* are placed around the subject, etc.), we derive the lexical scheme for the noun group  $\tilde{S}$  based on regular expressions:

$$\tilde{S} = ([A]\{0, n\}[S]\{1, m\}[P]), \quad (7)$$

where  $A = a_1 a_2 a_3 \dots a_{N-1} a_N$  is a sequence of adjectives, and the entry  $[A]\{0, n\}$  is a selection from 0 to  $n$  adjectives from  $a_1 a_2 a_3 \dots a_{N-1} a_N$ , at  $n < N$ ;  $S = s_1 s_2 s_3 \dots s_{M-1} s_M$  is a sequence of nouns, and the entry  $[S]\{1, m\}$  is a selection from 1 to  $m$  nouns from  $s_1 s_2 s_3 \dots s_{M-1} s_M$ , at  $m < M$ ;  $P = p_1 p_2 p_3 \dots p_{K-1} p_K$  is a sequence of pronouns, and the entry  $[P]$  is the choice of 1 pronoun from  $p_1 p_2 p_3 \dots p_{K-1} p_K$ ; record  $(x|y)$  is a choice of either  $x$ , or  $y$ ; the values of  $a_i$  and  $s_j$  agree in gender, number and case. Accordingly, for the *verb group*, the lexical scheme based on RE-expressions:

$$\tilde{V} = ([V]\{1, n\}[\tilde{S}']\{0, m\}[\tilde{S}']\{0, m\}[V]\{1, n\}), \quad (8)$$

where  $V = v_1 v_2 v_3 \dots v_{N-1} v_N$  is a sequence of verbs, and the entry  $[V]\{1, n\}$  is a choice from 1 to  $n$  verbs from  $v_1 v_2 v_3 \dots v_{N-1} v_N$ , at  $n < N$ ;  $\tilde{S}' = \tilde{S}_1 \tilde{S}_2 \tilde{S}_3 \dots \tilde{S}_{M-1} \tilde{S}_M$  is a sequence of noun groups, and the entry  $[\tilde{S}']\{0, m\}$  is a choice from 0 to  $m$  noun groups from  $\tilde{S}_1 \tilde{S}_2 \tilde{S}_3 \dots \tilde{S}_{M-1} \tilde{S}_M$ , at  $m < M$ ; entry  $(x|y)$  is choice of either  $x$ , or  $y$ ; agreement between  $v_i$  and  $\tilde{S}_j$  is carried out by person, gender and number. The lexical scheme of a *Ukrainian sentence* based on RE-expressions:

$$R = ([\tilde{S}']\{0, 1\}[\tilde{V}']\{0, 1\}[\tilde{V}']\{0, 1\}[\tilde{S}']\{0, 1\}), \quad (9)$$

where  $\tilde{V}' = \tilde{V}_1 \tilde{V}_2 \tilde{V}_3 \dots \tilde{V}_{N-1} \tilde{V}_N$  is a sequence of verb groups, and the entry  $[\tilde{V}']\{0, 1\}$  is a selection from 0 to 1 verb groups with  $\tilde{V}_1 \tilde{V}_2 \tilde{V}_3 \dots \tilde{V}_{N-1} \tilde{V}_N$  with the presence of a predicate;

$\tilde{S}' = \tilde{S}_1\tilde{S}_2\tilde{S}_3 \dots \tilde{S}_{M-1}\tilde{S}_M$  is a sequence of noun groups, and the entry  $[\tilde{S}']\{0,1\}$  is a selection from 0 to 1 noun groups from  $\tilde{S}_1\tilde{S}_2\tilde{S}_3 \dots \tilde{S}_{M-1}\tilde{S}_M$  with the presence of a subject; record  $(x|y)$  is a choice of  $x$  or  $y$ ; agreement between  $\tilde{V}_i$  and  $\tilde{S}_j$  is carried out by person, gender and number.

The main lexical features of the verb group are tense, number, person. For comparison, the lexical scheme of the noun group based on the RE-expression for an English-language sentence:

$$\tilde{S} = (\text{article}[A]\{0, n\}[S]/\text{of}[A]\{0, n\}[S]/\{0, m\}[[P]]). \quad (10)$$

The lexical scheme of the English verb group based on the RE-expression:

$$\tilde{V} = [V][\tilde{S}']\{0, m\}. \quad (11)$$

Lexical scheme for an English-language sentence based on the RE-expression:

$$R = [\tilde{S}'][\tilde{V}']. \quad (12)$$

The agreement of cases between the lexical units of the Ukrainian-language sentence affects the further syntactic and semantic analysis of the content:

$$\left. \begin{array}{l} 1. R \rightarrow RY_i x'_i, \\ 2. x'_i Y_j \rightarrow Y_j x'_i, \\ 3. RY_i \rightarrow x'_i R, \\ 4. R \rightarrow q. \end{array} \right\} i, j = 1, 2, 3, \quad (13)$$

where  $x_i, x'_i, q$  are the main lexical units;  $R, Y_i$  are auxiliary lexical units;  $R$  is the initial symbol as an indicator of the type of sentence chain generation.

Stages of lexical formation of a chain of tokens  $x_2 x_1 x_1 x_3 q x'_2 x'_1 x'_1 x'_3$ :





- |   |   |
|---|---|
| 1. $R$  | 6. (2) $RY_2 Y_1 x'_2 x'_1 Y_1 x'_1 Y_3 x'_3$   |
| 2. (1) $RY_3 x'_3$                            | 7. - 11. .... (2; 5 times)                      |
| 3. (1) $RY_1 x'_1 Y_3 x'_3$                   | 12. (3) $x_2 RY_1 Y_1 Y_3 x'_2 x'_1 x'_1 x'_3$  |
| 4. (1) $RY_1 x'_1 Y_1 x'_1 Y_3 x'_3$          | 13. - 15. .... (3; 3 times)                     |
| 5. (1) $RY_2 x'_2 Y_1 x'_1 Y_1 x'_1 Y_3 x'_3$ | 16. (4) $x_2 x_1 x_1 x_3 q x'_2 x'_1 x'_1 x'_3$ |

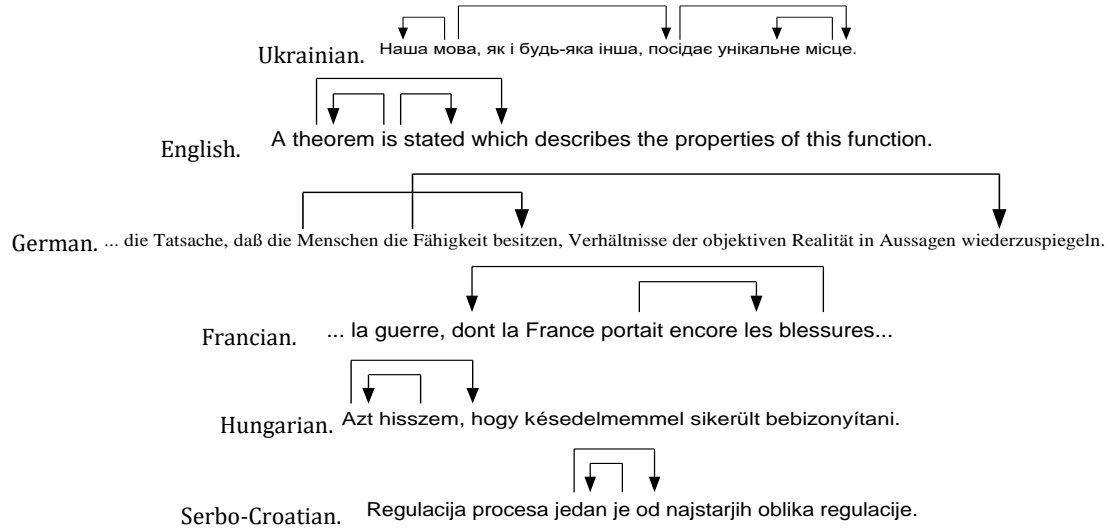
An example of lexical generation of the type  $\{xqx'\}$ :  $\overset{a}{\text{Саша}}, \overset{b}{\text{Софія}}, \overset{c}{\text{Катя}}, \overset{d}{\text{Данило}}, \dots - \overset{a'}{\text{спортсмен}}, \overset{b'}{\text{співачка}}, \overset{c'}{\text{художниця}}, \overset{d'}{\text{поет}}, \dots$  respectively, where  $x$  ( $abcd\dots$ ) is a sequence of proper names,  $x'$  ( $a'b'c'd'\dots$ ) is a sequence of professions agreed with proper names;  $q$  is a dash. Any verb has the ability to act as a complement: *моя дитина вподобала читання книг* [*moja dytyna vpodobala knyhochytannya*] [*My child liked reading books*]. This process can theoretically be repeated an unlimited number of times: *він читав цікаво, думає про читання книг* [*vin knyhochytannyatsikavodumaye pro knyhochytannyatsikavist'*] [*it is interesting to read books, thinks about reading books, interesting*], i.e.

$$\text{Він} \overbrace{\text{книг}}^a \overbrace{\text{читання}}^b \overbrace{\text{цікавість}}^c - \text{думає про} - \overbrace{\text{книг}}^{a'} \overbrace{\text{читання}}^{b'} \overbrace{\text{цікавість}}^{c'}.$$

A language consisting of strings of the form  $abcd\dots d'c'b'a'$  (composed of symbols  $a_1, a_2, a_3, a'_1, a'_2, a'_3$ ) is generated by a grammar of 6 rules:

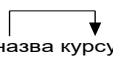
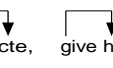




$$\left. \begin{array}{l} I \rightarrow a_i I a_i' \\ I \rightarrow a_i a_i' \end{array} \right\} i = 1, 2, 3. \quad (14)$$

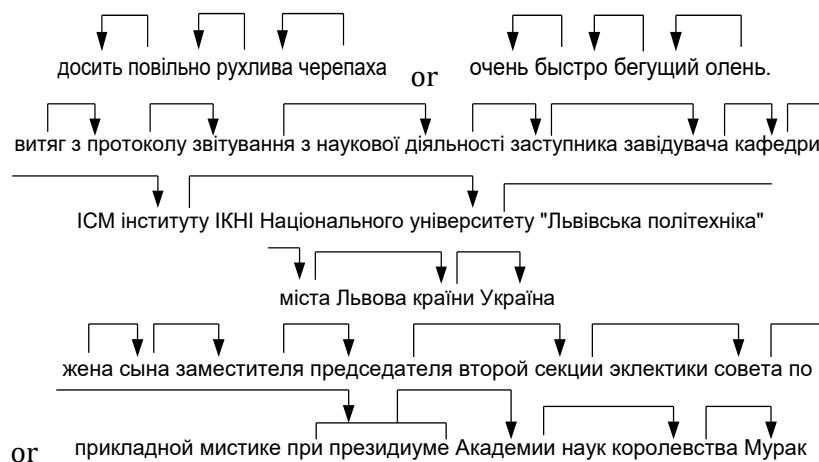
Such grammar do not provide, for example, a natural description for the so-called non-project constructions with breaks, crossing (  ,  ) or framing (  ,  ) directions of syntactic dependence (Fig. 19).



**Figure 19:** Examples of natural description for so-called non-design constructions

To describe such constructions of sentences are used:

1. Right subordination:     
2. Left submission:    
3. Sequential subordination (Fig. 20):



**Figure 20:** Examples of natural description of sequential subordination



$S_{c,y,z} \rightarrow \text{місто}_{y,z}, \dots, A_{x,y,z} \rightarrow \text{веселий}_{x,y,z}, \text{запальний}_{x,y,z}, \text{дитячий}_{x,y,z}, \dots \}$

Another derivation is to use more memory, such as starting derivation with  $\tilde{S}_{c,od,n} \rightarrow$   
*дуже*  $A_{c,od,n}$   $A_{c,od,n}$   $S_{c,od,n}$   $S_{c,od,n}$   $S_{ж,od,p}$   $S_{c,od,p}$   $S_{c,od,p}$

$\tilde{S}_{c,od,n}$   
 $A_{c,od,n} \tilde{S}_{c,od,n}$   
*дуже*  $A_{c,od,n} \tilde{S}_{c,od,n}$   
*дуже веселий*  $A_{c,od,n} \tilde{S}_{c,od,n}$   
*дуже веселий запальний*  $A_{c,od,n} \tilde{S}_{c,od,n}$   
*дуже веселий запальний дитячий*  $\tilde{S}_{c,od,n} \tilde{S}_{c,od,p}$   
*дуже веселий запальний дитячий*  $S_{c,od,n} \tilde{S}_{c,od,p}$   
*дуже веселий запальний дитячий сміх*  $\tilde{S}_{c,od,p}$   
*дуже веселий запальний дитячий сміх*  $\tilde{S}_{c,od,p} \tilde{S}_{ж,od,p}$   
*дуже веселий запальний дитячий сміх*  $S_{c,od,p} \tilde{S}_{ж,od,p}$   
*дуже веселий запальний дитячий сміх школяра*  $\tilde{S}_{ж,od,p}$   
*дуже веселий запальний дитячий сміх школяра*  $\tilde{S}_{ж,od,p} \tilde{S}_{c,od,p}$   
*дуже веселий запальний дитячий сміх школяра*  $S_{ж,od,p} \tilde{S}_{c,od,p}$   
*дуже веселий запальний дитячий сміх школяра школи*  $\tilde{S}_{c,od,p}$   
*дуже веселий запальний дитячий сміх школяра школи*  $\tilde{S}_{c,od,p} \tilde{S}_{c,od,p}$   
*дуже веселий запальний дитячий сміх школяра школи*  $S_{c,od,p} \tilde{S}_{c,od,p}$   
*дуже веселий запальний дитячий сміх школяра школи міста*  $\tilde{S}_{c,od,p}$   
*дуже веселий запальний дитячий сміх школяра школи міста*  $S_{c,od,p}$   
*дуже веселий запальний дитячий сміх школяра школи міста Львова*

**Figure 22:** The process of deriving the Ukrainian-language chain for example 2

Kivánom, hogy valamint az agyag<sup>23</sup> cilelx karjai<sup>22</sup> közül kibontakozni<sup>21</sup> akary<sup>20</sup>  
kocsikerék<sup>19</sup> rettentx nyikorgósbtyl<sup>18</sup> megriadt<sup>17</sup> juhószkutya<sup>16</sup> bundójbba<sup>15</sup>  
kapaszokody<sup>14</sup> kullancs<sup>13</sup> kidülledt fílszeméibx<sup>12</sup> albóseppent<sup>11</sup> kunnyeseppben<sup>10</sup>  
visszatzkruzdxdx<sup>9</sup> holdvilág fínyítxl<sup>8</sup> illuminólt<sup>7</sup> rablylovagvóbr<sup>6</sup> felvonyhidjóbbyl<sup>5</sup>  
kiólylly<sup>4</sup> vasszegek<sup>3</sup> kohíziys erejéinek<sup>2</sup> hatósa<sup>1</sup> évszázadokra összetartja annak  
materiáját, aképpen tartsa össze ezt a társaságot az igaz szeretet.  
Я хочу, аби справжнє кохання скріпило цю компанію так, як на століття  
скріплює матеріал мосту дія<sup>1</sup> єднальної сили<sup>2</sup> цвяхів<sup>3</sup>, що торчать<sup>4</sup> з  
підіймального мосту<sup>5</sup> розбійницького феодального замку<sup>6</sup>, освященного<sup>7</sup>  
місячним світлом<sup>8</sup>, що відображається<sup>9</sup> в краплині<sup>10</sup>, яка витікає<sup>11</sup> з витрішеного  
ока<sup>12</sup> клеца<sup>13</sup>, що вчепилася<sup>14</sup> в шерсть<sup>15</sup> вівчарки<sup>16</sup>, наполоханої<sup>17</sup> жахливим  
скрипом<sup>18</sup> возових колес<sup>19</sup>, що прагнуть<sup>20</sup> вирватися<sup>21</sup> з обіймів<sup>22</sup> грязюки<sup>23</sup>.

**Figure 23:** An example from H. Feher’s short story – A Humorous Toast

There are cases in the textual content when not only the right but also the left sequential subordination has an unlimited depth of derivation, for example, due to subordinate clauses with the operative word *which*, *what*, *when*, etc. (*тваринка, яку врятувала Софія* [tvarynka, yaku vryatuvava Sofiya] - the animal that Sofia saved). Fig. 23 illustrates a phrase with a depth of 22 and is completely grammatically correct (as is its Ukrainian version). Moreover, nothing prevents you from continuing the phrase to the left *на волю в обійми зеленої пахучої трави* [na volyu v obiyumu zelenoyi pakhuchoyi travy] (freely into the embrace of green, fragrant grass). The Ukrainian language allows you to generate phrases with an unlimited number of sequentially subordinating from left to right constructions of the type  $Y_1 Y_2 \dots Y_i \dots$  (unlimited right subordination), and at the same time, unlimited left

subordination is possible in each of the constructions  $X_i$  - a sequence of chains  $\dots Y_{ij} \dots Y_{i3} Y_{i2} Y_{i1}$ ; however, within the sequence  $Y_{ij}$  further unlimited expansion is impossible. According to the rules of the Ukrainian language  $Y_i$  are interpreted as simple sentences, each of which is an additional determiner to the previous one, and  $Y_{ij}$  are interpreted as prepositive adjective inflexions.

The grammar  $G' = \langle D', D'_1, I', R' \rangle$  has a basic dictionary  $D' = N_1, N_2, \dots, N_n$  symbols and rules of the form  $R' = \{Y \rightarrow ZN_i, X \rightarrow N_i\}$ , where  $Y \in D'_1$  and  $Z \in D'_1$ . Each of  $N_i$  corresponds to some regular grammar  $G'_i = \langle D, D'_1, N_i, R_i \rangle$ , where  $D$  is the main dictionary  $\forall G'_i$ ,  $D'_1$  is the auxiliary dictionary for  $D'_1 \cap D' = N_i$  and  $D'_1 \cap D'_1 = N_i$ ;  $N_i$  is the initial symbol; scheme rules of the form  $R_i = \{C \rightarrow eE, C \rightarrow c\}$  (heading Latin characters are non-terminal, and line characters are terminal). The non-terminal dictionaries of the grammar  $G'_i$  are pairwise disjoint. Association:

$$G = G' \cup G'_1 \cup G'_2 \cup \dots \cup G'_n, \quad (15)$$

where the main dictionary  $D$  in all  $G'_i$ , and the auxiliary additional dictionary and scheme:

$$D_1 = D' \cup D'_1 \cup D_1^1 \cup D_1^2 \cup \dots \cup D_1^n, \quad R = R' \cup R_1 \cup R_2 \cup \dots \cup R_n \quad (16)$$

The grammar  $G$  is special and equivalent to an automatic one, for example:

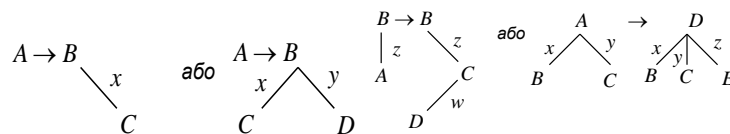
$$R' = \begin{cases} I \rightarrow BN_1 \\ B \rightarrow CN_1 \\ C \rightarrow BN_2 \\ C \rightarrow EN_3 \\ E \rightarrow EN_4 \\ E \rightarrow N_2 \end{cases}, R_1 = \begin{cases} N_1 \rightarrow bP_1 \\ P_1 \rightarrow aQ_1 \\ Q_1 \rightarrow aQ_1 \\ Q_1 \rightarrow c \end{cases}, R_2 = \{N_2 \rightarrow d, R_3 = \begin{cases} N_3 \rightarrow aP_3 \\ N_3 \rightarrow bQ_3 \\ N_3 \rightarrow cW_3 \\ P_3 \rightarrow a \\ Q_3 \rightarrow b \\ W_3 \rightarrow dW_3 \\ W_3 \rightarrow eW_3 \\ W_3 \rightarrow d \end{cases}, R_4 = \begin{cases} N_4 \rightarrow cP_4 \\ P_4 \rightarrow b \end{cases}, \quad (17)$$

**Algorithm 4.3. Algorithm of sentence syntactic analysis.**

**Stage 1.** An unconstrained generated sequence is generated to the right by  $N_i$  as a syntactic group or sentence based on the rules of  $R'$ .

**Stage 2.** Any of  $N_i$  based on  $R_i$  is expanded indefinitely in the form of a tree (Fig. 24) from right to left - into a chain of terminal symbols as words.

To analyze the syntactic structure of a sentence is to identify the order of words depending on the syntactic structure and relationships, which is determined necessarily according to the analysis of neighbours and something derived/secondary. It is advisable to modify the grammar so that both parts of the predicate (Fig. 24) are trees of syntactic relations. Lines with subscripts describe syntactic relations of various types; symbols  $A, B, C, \dots$  are syntactic categories.



**Figure 24:** Rules for building a tree

As a result, the syntactic structures (rather than phrases) of the language are obtained as part of the generative grammar. Another part of this grammar is the calculation in the Ukrainian language - with mandatory consideration of the logical derivation of linear sequences of words, solving the problem of discontinuous constituents.

#### 4.5. The method of semantic analysis of the Ukrainian language

Semantic analysis consists not only in identifying the content of the text but also in generating data structures to which logical reasoning can be applied. Thematic Meaning Representations (TMR) are used to encode sentences in the form of predicate structures based on first-order logic or lambda calculus ( $\lambda$ -calculus). Network/graph structures are used to encode interactions of predicates of relevant text features. Then a traversal is implemented to analyze the centrality of terms or subjects and the reasons for the relationships between elements.

Analysis of graphs, including ontology  $O$ , is usually not a complete SEM, but helps to form part of important logical decisions/conclusions based on the taxonomy of concepts  $X$ :

$$O: ULSR \rightarrow Concepts. \quad (18)$$

The result of SEM based on the ontological model of the rules of the syntax of the Ukrainian language  $O$  are weighted oriented graphs of the semantics of the text:

$$O = \langle Concepts, Relationships, Functions \rangle, \quad (19)$$

where *Relationships* is a tuple of relationships between SA concepts of the Ukrainian language; *Concepts* is a tuple of SA concepts describing the rules of the Ukrainian language; *Functions* is a tuple of functions for the interpretation of concepts/rules of the Ukrainian language.

The taxonomy of concepts sets the syntax of the language as the root concept of the ontology:

$$Concepts_{\mu}: \langle R_{Snt} \rangle \rightarrow C'_{\mu}. \quad (20)$$

The optimal definition of the tuple of relations between these concepts and the tuple of the rules of the Ukrainian language, formalized by the descriptive logic of DL, will allow effective processing of Ukrainian texts:

$$Concepts = \langle R_{Mrp}, R_{Pnc}, R_{Str}, R_{Snt}, R_{Smn} \rangle, \quad (21)$$

where tuples of concepts of morphology  $R_{Mrp}$ , punctuation  $R_{Pnc}$ , structure  $R_{Str}$ , syntax  $R_{Snt}$  (Fig. 25) and semantics  $R_{Smn}$ .

In SEM, to identify the set of semes of the corresponding text and their relationship, first, based on the results of SYA, a semantic graph of the relations of linguistic units is built, taking into account the parts of the language of words:

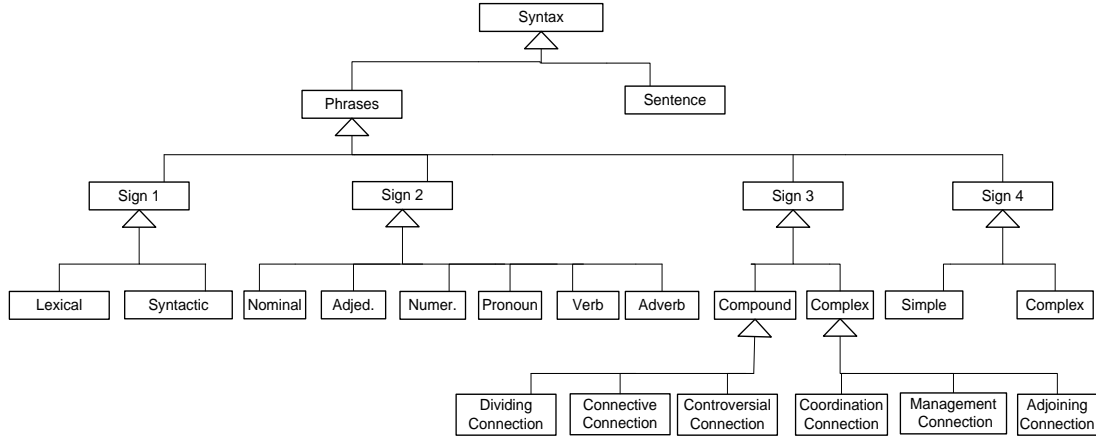
$$C'_{\mu} = \lambda(C_{\lambda}, D_{\lambda}, R_{\lambda}, Concepts_{\mu}), \quad Concepts_{\mu} = \langle C_{WrdCmb}, C_{SntCmb} \rangle, \quad (22)$$

where  $C_{WrdCmb}$  is a tuple of word formation concepts;  $C_{SntCmb}$  is a tuple of sentence generation concepts in the Ukrainian language (Fig. 26).

Tuple  $C_{WrdCmb}$  according to the rules of the Ukrainian language syntax (Fig. 26):

$$C_{WrdCmb} = \langle Sgn_1^{Wrd}, Sgn_2^{Wrd}, Sgn_3^{Wrd}, Sgn_4^{Wrd} \rangle, \quad (23)$$

where  $Sgn_i^{Wrd}$  is a tuple of phrase generation properties.

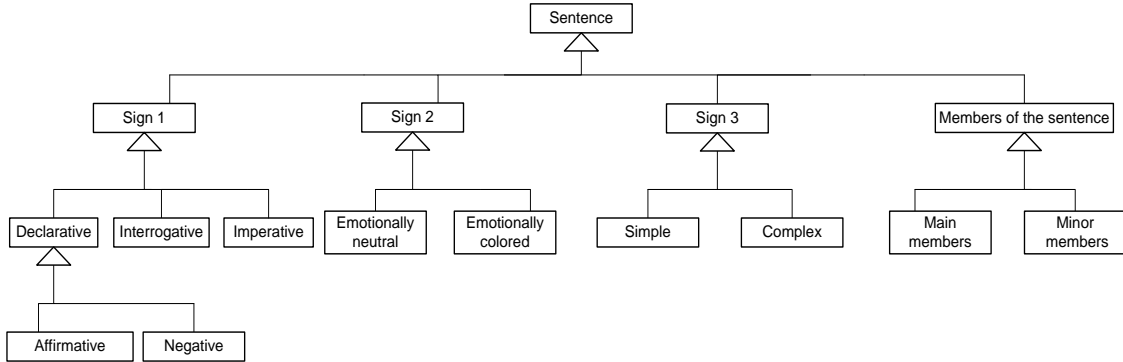


**Figure 25:** Class diagram for the Syntax Phrase type hierarchy

The tuple  $Sgn_1^{Wrd}$  according to the rules of the Ukrainian language syntax (Fig. 26):

$$Sgn_1^{Wrd} = \langle Sgn_{Lxc}^I, Sgn_{Snt}^I \rangle, \quad (24)$$

where  $Sgn_{Lxc}^I$  is a tuple of lexical features of phrase generation;  $Sgn_{Snt}^I$  is a tuple of syntactic signs of phrase generation.



**Figure 26:** Class diagram for the sentence type hierarchy

$$Sgn_2^{Wrd} = \langle Sgn_{Nou}^{II}, Sgn_{Adc}^{II}, Sgn_{Nmr}^{II}, Sgn_{Prn}^{II}, Sgn_{Vrb}^{II}, Sgn_{Adv}^{II} \rangle, \quad (25)$$

where  $Sgn_{Nou}^{II}$  is a tuple of named properties;  $Sgn_{Adc}^{II}$  is a tuple of adjectival properties;  $Sgn_{Nmr}^{II}$  is a tuple of numerical properties;  $Sgn_{Prn}^{II}$  is a tuple of pronominal properties;  $Sgn_{Vrb}^{II}$  is a tuple of verb properties;  $Sgn_{Adv}^{II}$  is a tuple of adverbial properties;

$$Sgn_3^{Wrd} = \langle Sgn_{Crd}^{III}, Sgn_{Inf}^{III} \rangle, \quad (26)$$

where  $Sgn_{Crd}^{III}$  is a tuple of consecutive and  $Sgn_{Inf}^{III}$  is a tuple of subordinate properties;

$$Sgn_4^{Wrd} = \langle Sgn_{SmWd}^{IV}, Sgn_{CmWd}^{IV} \rangle, \quad (27)$$



where  $Sgn_{SmWd}^{IV}$  is a tuple of simple properties and  $Sgn_{CmWd}^{IV}$  is a tuple of complex properties. The tuple  $Sgn_{Crd}^{III}$  describes the component properties of the relation sentence:

$$Sgn_{Crd}^{III} = \langle Sgn_{AdCm}^{Crd}, Sgn_{CnCm}^{Crd}, Sgn_{DvCm}^{Crd} \rangle, \quad (28)$$

where  $Sgn_{AdCm}^{Crd}$  is a tuple of properties of separating,  $Sgn_{CnCm}^{Crd}$  is a tuple of properties of connecting and  $Sgn_{DvCm}^{Crd}$  is a tuple of properties of opposite connections.

$$Sgn_{Inf}^{III} = \langle Sgn_{CtCm}^{Inf}, Sgn_{MgCm}^{Inf}, Sgn_{AgCm}^{Inf} \rangle. \quad (29)$$

where  $Sgn_{CtCm}^{Inf}$  is a tuple of matching properties;  $Sgn_{MgCm}^{Inf}$  is a tuple of control properties;  $Sgn_{AgCm}^{Inf}$  is a tuple of fit properties.

A tuple of sentence generation concepts in the Ukrainian language (Fig. 26):

$$C_{SntCmb} = \langle Sgn_1^{Snt}, Sgn_2^{Snt}, Sgn_3^{Snt}, Sgn_{SnMb}^{Snt} \rangle, \quad (30)$$

where sentence generation properties in the Ukrainian language are grouped in  $Sgn_i^{Snt}$  is tuples of sentence generation properties in the Ukrainian language;  $Sgn_{SnMb}^{Snt}$  is a tuple of properties of identification of sentence members;

$$Sgn_1^{Snt} = \langle Sgn_{NrSn}^I, Sgn_{PrSn}^I, Sgn_{InSn}^I \rangle, \quad (31)$$

where  $Sgn_{NrSn}^I$  is a tuple of narrative sentence generation properties;  $Sgn_{PrSn}^I$  is a tuple of properties of generating interrogative sentences;  $Sgn_{InSn}^I$  is a tuple of properties of generating motivational sentences;

$$Sgn_2^{Snt} = \langle Sgn_{EmNt}^{II}, Sgn_{EmCl}^{II} \rangle, \quad (32)$$

where  $Sgn_{EmNt}^{II}$  is a tuple of properties for generating emotionally neutral sentences;  $Sgn_{EmCl}^{II}$  is a tuple of properties for generating emotionally coloured sentences;

$$Sgn_3^{Snt} = \langle Sgn_{SlSt}^{III}, Sgn_{ClSt}^{III} \rangle, \quad (33)$$

where a tuple of concepts for the formation of  $Sgn_{SlSt}^{III}$  simple and  $Sgn_{ClSt}^{III}$  complex sentences;

$$Sgn_{SnMb}^{Snt} = \langle Sgn_{MnStMb}^{SnMb}, Sgn_{SdStMb}^{SnMb} \rangle, \quad (34)$$

where  $Sgn_{MnStMb}^{SnMb}$  is a tuple of properties identifying the main members of the sentence;  $Sgn_{SdStMb}^{SnMb}$  is a tuple of properties of identification of secondary members of the sentence;

$$Sgn_{NrSn}^I = \langle Sgn_{AfSt}^{NrSn}, Sgn_{NgSt}^{NrSn} \rangle, \quad (35)$$

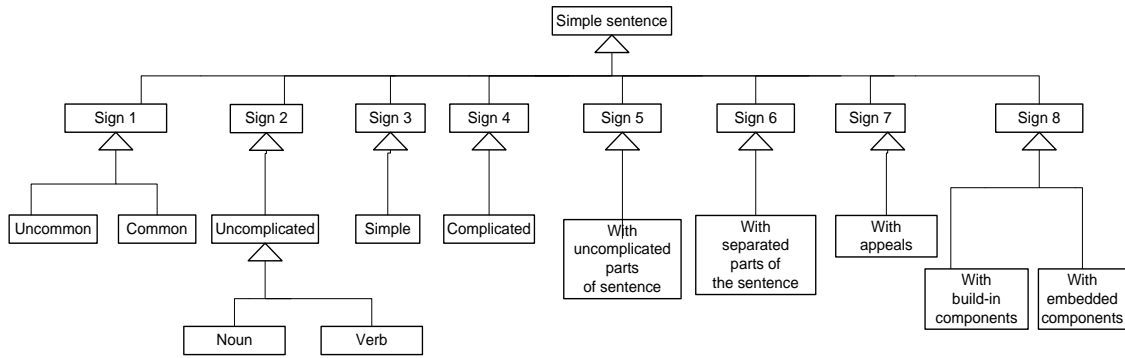
where  $Sgn_{AfSt}^{NrSn}$  is a tuple of properties of generating affirmative sentences;  $Sgn_{NgSt}^{NrSn}$  is a tuple of negative sentence generation properties.

To generate a simple sentence  $Sgn_{SlSt}^{III}$ , the signs are analyzed (Fig. 27):

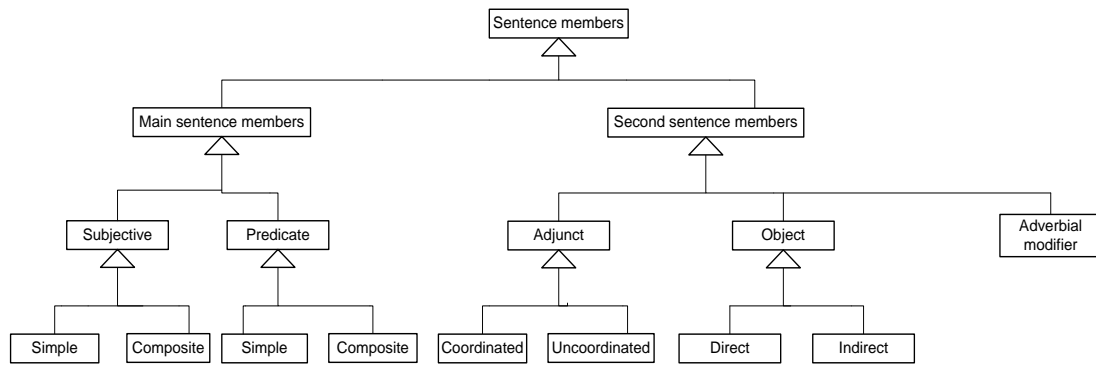
$$Sgn_{SlSt}^{III} = \langle Sgn_1^{SlSt}, Sgn_2^{SlSt}, Sgn_3^{SlSt}, Sgn_4^{SlSt}, Sgn_5^{SlSt}, Sgn_6^{SlSt}, Sgn_7^{SlSt}, Sgn_8^{SlSt} \rangle, \quad (36)$$

where  $Sgn_i^{SlSt}$  is a tuple of simple sentence generation properties.

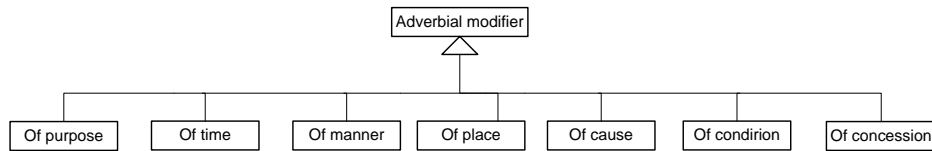
Similarly, tuples are formed to identify the members of the sentence  $Sgn_{SnMb}^{Snt}$  (Fig. 28- Fig. 29) and the complex sentence  $Sgn_{ClSt}^{III}$  (Fig. 30).



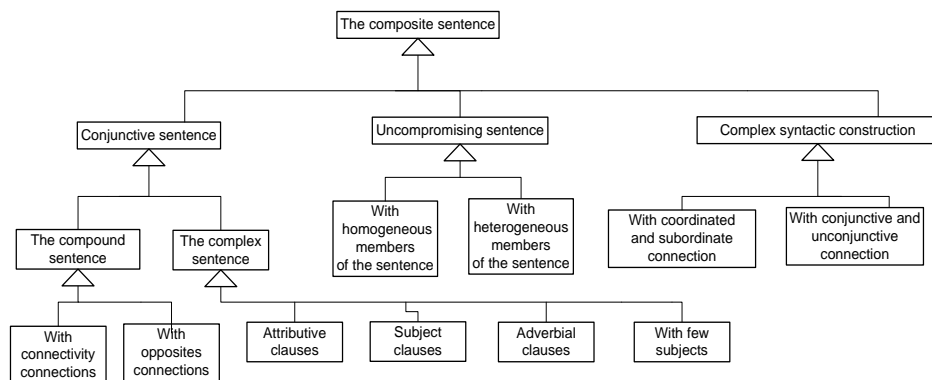
**Figure 27:** Class diagram for a hierarchy of the type Simple sentence



**Figure 28:** Class diagram for a hierarchy of the type the Sentence Members



**Figure 29:** Class diagram for the Circumstance type hierarchy



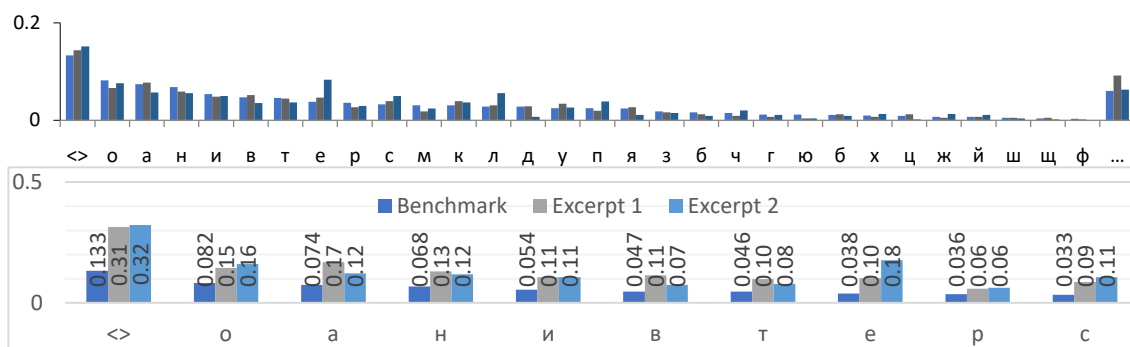
**Figure 30:** Class diagram for the Complex Sentence type hierarchy

The process of extracting data from the Ukrainian-language text based on the syntax ontology allows you to supplement the conceptual weighting graphs of the content.

#### **4.6. The method of pragmatic analysis of the Ukrainian language**

Pragmatics examines the dependence of meaning on the context of the textual content of the author and takes into account his prior knowledge, intentions, purpose, etc., in contrast to semantics, which analyzes the meaning itself depending on the results of GA, MA, LA and SYA within a particular text. Pragmatics is a continuation of SEM, taking into account the peculiarities of the context of the analysed text, taking into account the ambiguity of the statements of the analyzed text, based on the analysis of the features of the author's statements in previous similar texts, based on the time, place, method, purpose and other circumstances of the conversation.

In PA, when resolving the ambiguity of the author's speech in a specific analyzed text, taking into account the features of the author's speech in previous similar speeches, it is best to use word prediction models, for example, N-grammatical Language Models (LM). Each speaker, as a person with a unique life experience, has not only his dictionary of thematic words but also a unique handwriting of the use of these words and their sequence in a certain context of the relevant thematic direction. In the expression «*лінгвістична система опрацьовує ...*» [linhvistychna systema oprats'ovuye ...] (the linguistic system processes ...) the next word depends not only on the context but also on the so-called speech handwriting of the author of the text: *текст, контент, текстовий контент, вхідні дані, вхідну інформацію, інтегровані дані, авторський контент, публікації* [tekst, kontent, tekstovyy kontent, vkhidni dani, vkhidnu informatsiyu, intehrovani dani, avtors'kyu kontent, publikatsiyi] (text, content, text content, input data, input information, integrated data, author content, publications), etc. The phrase «*включіть свою виконану лабораторну роботу ...*» [vklyuchit' svoyu vykonanu laboratornu robotu ...] (include your completed lab work...) as opposed to «*додайте свою виконану лабораторну роботу ...*» [dodayte svoyu vykonanu laboratornu robotu ...] (add your completed lab work...) has a broader meaning and depends significantly not only on the context but also on the speaker (include can mean like download the developed software on the computer or in the sense of adding it as an item to some list, etc.). Dialogue participants intuitively understand the content based on their experience of communicating with the author of the phrase. Pragmatic analysis requires the introduction of models that determine the probability for each subsequent word. They are also intended for assigning the probability of the target utterance for correct machine translation, identification/correction of grammatical and stylistic errors, and handwriting or language recognition. Each language has special statistical parameters, and the analysis of the probability of the appearance of only letters and their combinations as N-grams of the corresponding language makes it possible to identify the language itself or the style of the author (Fig. 31 - with greater probability, the author of the benchmark wrote Excerpt 1).



**Figure 31:** Probability of appearance of letters in the standard and analyzed passages

For Ukrainian texts, the statistical parameters of styles are the probabilities of vowels, consonants, and gaps between words, as well as soft and sonorous groups of consonants. Probability is also important for enhancing communication. Physicist Stephen Hawking used simple movements to select words from a menu for speech synthesis. For such IS, it is appropriate to use word prediction to generate suggestions for a list of likely words for the menu. One of the most widespread and easiest to implement for English-language texts is LM - N-gram, which assigns probabilities to sentences or sequences of words. For Ukrainian-language texts, it is better to apply such LM to the sequence of word bases without taking into account inflexions (otherwise incorrect PA results will be obtained) to calculate  $P(b|a)$  is the probability of the appearance of the base of the word  $b$  after the sequence of bases  $a$ . Taking into account words in N-grams of LM in Ukrainian-language texts is appropriate for identifying grammatical errors.

$$P(\text{систем}|\text{комп'ютер лінгвіст}), P(\text{систему}|\text{комп'ютерні лінгвістичні}), \quad (37)$$

$$P(\text{систему}|\text{комп'ютерну лінгвістичну}).$$

One of the best ways to calculate such a probability is to conduct a statistical analysis on large corpora of texts of the relevant author or relevant thematic direction from reliable Internet sources.:

$$P(\text{систем}|\text{комп'ютер лінгвіст}) = \frac{N(\text{комп'ютер лінгвіст систем})}{N(\text{комп'ютер лінгвіст})}, \quad (38)$$

$$P(\text{систем}|\text{комп'ютер лінгвіст}) = \frac{P(\text{комп'ютер лінгвіст систем})}{P(\text{комп'ютер лінгвіст})}.$$

This gives a probabilistic result for a certain period because the language is creative, not homogeneous, and the vocabulary is updated and constantly develops both in general and for a specific speaker - the author of the text. To analyze the corresponding random linguistic event  $A_i = \text{комп'ютер}$ ,  $P(A_i)$  is found to calculate the probability of the appearance of a certain sequence of linguistic events based on the chain rule or the general product rule (chain rule of probability):

$$P(A_1 A_2 \dots A_n) = P(A_1) P(A_2 | A_1) P(A_3 | A_1^2) \dots P(A_n | A_1^{n-1}), \quad (39)$$

$$P(A_1 A_2 \dots A_n) = \prod_{i=1}^n P(A_i | A_1^{i-1}).$$

To analyze the sequence of  $N$  bases of words  $x_1 x_2 \dots x_n$  or  $x_1^n$  ( $x_1 x_2 \dots x_{n-1} \rightarrow x_1^{n-1}$ ) when  $A_1 = x_1, A_2 = x_2, A_3 = x_3, \dots, A_n = x_n$  calculate:

$$P(x_1 x_2 \dots x_n) = P(x_1^n) = P(x_1)P(x_2|x_1)P(x_3|x_1^2) \dots P(x_n|x_1^{n-1}), \quad (40)$$

$$P(x_1^n) = \prod_{i=1}^n P(x_i|x_1^{i-1}). \quad (41)$$

The chain rule reflects the relationship between the overall probability of the appearance of a specific sequence of bases and the conditional probability of the appearance of a word base by specific previous word bases in this sequence. Taking into account the entire dynamics of the occurrence of all word bases in the text to sequences of other word bases is a redundant/inefficient process due to the variability of language/speech over time. Prediction of the 2-gram model consists of approximating the dynamics of the appearance of only the last few bases of words in a given sequence:

$$P(\text{систем}|\text{лінгвіст}) = \frac{N(\text{лінгвіст систем})}{N(\text{лінгвіст})}, \quad P(\text{систем}|\text{лінгвіст}) = \frac{P(\text{лінгвіст систем})}{P(\text{лінгвіст})} \quad (42)$$

To forecast the conditional probability of the following base of the word, we use the Markov assumption (the probability of the word depends only on the previous one):

$$P(x_n|x_1^{n-1}) \approx P(x_n|x_{n-1}). \quad (43)$$

To predict the conditional probability of the next base of the word in the N-gram based on the metric of Maximum (greatest) Likelihood Estimation (MLE) we calculate:

$$P(x_n|x_1^{n-1}) \approx P(x_n|x_{n-k+1}^{n-1}). \quad (44)$$

Based on this, we calculate the probability of a complete sequence of word stems:

$$P(x_1^n) \approx \prod_{i=1}^n P(x_i|x_{i-1}), \quad (45)$$

We find the MLE estimate for the parameters of the N-gram model by statistically analyzing the corresponding text corpus and normalizing the frequency of occurrences of word bases and their sequences within  $[0;1]$ :

$$P(x_n|x_{n-1}) = \frac{N(x_{n-1}x_n)}{\sum_x N(x_{n-1}x)} = \frac{N(x_{n-1}x_n)}{N(x_{n-1})}. \quad (46)$$

For example, for three sentences of the mini-corpus (conditionally, the  $\langle p \rangle \langle /p \rangle$  tags are the boundaries of one sentence), we will calculate the Markov assumption of the 2-gram occurrence of word bases:

$\langle p \rangle$  CLS опрацьовує текстовий контент на основі NLP-процесів  $\langle /p \rangle$   
 $\langle p \rangle$  Інтеграція текстового контенту є одним із основних процесів CLS  $\langle /p \rangle$   
 $\langle p \rangle$  CLS розв'язує конкретну NLP-задачу для відповідного контенту  $\langle /p \rangle$

$$P(\text{CLS} | \langle p \rangle) = \frac{2}{3}; P(\text{інтегр} | \langle p \rangle) = \frac{1}{3}; P(\text{опрац} | \text{CLS}) = \frac{1}{3};$$

$$P(\langle /p \rangle | \text{контент}) = \frac{1}{3}; P(\text{контент} | \text{текст}) = \frac{2}{3}; P(\text{задач} | \text{NLP}) = \frac{1}{2}.$$

Estimation of the MLE parameter for the N-gram model as a relative frequency:

$$P(x_n | x_{n-k+1}^{n-1}) = \frac{N(x_{n-k+1}^{n-1} x_n)}{N(x_{n-k+1}^{n-1})} \quad (47)$$

**Algorithm 4.4.** Algorithm for the analysis of MLE-parameter estimates for the N-gram model.

**Stage 1.** Parse the input text and break it into separate phrases (sentences)  $R_1 R_2 \dots R_m$ , marking each start-end with a corresponding  $\langle p \rangle \langle /p \rangle$  tag. Eliminate all non-alphabetic characters. Convert uppercase letters to lowercase. Remove service words if necessary (for certain NLP tasks).

**Stage 2.** Apply Porter's stemming to obtain the sequence of word bases  $x_{i1} x_{i2} \dots x_{in_i}$  of word bases  $\forall R_i$  taking into account word normalization.

**Stage 3.** Receive input requests  $Q_1 Q_2 \dots Q_k$  as a sequence of words of the searched data. Find  $\forall Q_j$  for each word  $y_{j1} y_{j2} \dots y_{jk_j}$  basis by stemming.

For example, for the search phrase  $Q_j$ :  
*Методи та засоби опрацювання інформаційних ресурсів систем електронної контент комерції*

$y_{j1}$	$y_{j2}$	$y_{j3}$	$y_{j4}$	$y_{j5}$	$y_{j6}$	$y_{j7}$	$y_{j8}$	$y_{j9}$	$y_{j10}$
метод	та	засіб	опрац	інформ	ресурс	систем	електрон	контент	комерц
58	190	25	62	122	83	170	89	408	300

**Stage 4.** Conduct a statistical analysis of the occurrence of word bases and sequences of query word bases in the analyzed text.

Basics of words of analyzed text	$x_{i1}$	$x_{i2}$	$x_{i3}$	$x_{i4}$	$x_{i5}$	$x_{i6}$	$x_{i7}$	$x_{i8}$	$x_{i9}$	$x_{i10}$
$x_{i1}$ метод	0	8	0	6	0	0	0	0	1	0
$x_{i2}$ та	2	0	5	1	7	0	2	0	0	1
$x_{i3}$ засіб	0	2	0	14	0	0	0	0	0	0
$x_{i4}$ опрац	0	0	0	0	46	0	0	1	3	4
$x_{i5}$ інформ	0	0	0	0	0	64	9	0	0	0
$x_{i6}$ ресурс	0	7	0	0	0	0	0	1	0	0
$x_{i7}$ систем	0	8	0	1	0	0	0	21	0	0
$x_{i8}$ електрон	0	0	0	0	0	0	0	0	72	10
$x_{i9}$ контент	0	10	0	0	0	0	0	0	0	73
$x_{i10}$ комерц	0	6	0	0	0	0	0	0	176	0

**Stage 5.** Find the probability of occurrence of 2-grams in the analyzed text. In each row, the value is divided by  $y_{ji}$ , where  $i$  is the row number after normalization.

Basics of words of analyzed text	$x_{i1}$	$x_{i2}$	$x_{i3}$	$x_{i4}$	$x_{i5}$	$x_{i6}$	$x_{i7}$	$x_{i8}$	$x_{i9}$	$x_{i10}$	$y_{ji}$
$x_{i1}$ метод	0	0.18	0	0.1	0	0	0	0	0.02	0	58
$x_{i2}$ та	0.01	0	0.03	0.005	0.035	0	0.01	0	0	0.005	190
$x_{i3}$ засіб	0	0.08	0	0.16	0	0	0	0	0	0	25
$x_{i4}$ опрац	0	0	0	0	0.74	0	0	0.016	0.048	0.064	62
$x_{i5}$ інформ	0	0	0	0	0	0.52	0.074	0	0	0	122
$x_{i6}$ ресурс	0	0.084	0	0	0	0	0	0.012	0	0	83
$x_{i7}$ систем	0	0.047	0	0.006	0	0	0	0.124	0	0	170
$x_{i8}$ електрон	0	0	0	0	0	0	0	0	0.81	0.112	89
$x_{i9}$ контент	0	0.025	0	0	0	0	0	0	0	0.179	408
$x_{i10}$ комерц	0	0.02	0	0	0	0	0	0	0.053	0	300

With each subsequent multiplication, the probability decreases. Applying the logarithm of probabilities (log probabilities) will allow you to operate with not-so-small values for calculating accuracy.

$$\prod_{i=1}^n P_i = e^{\sum_{i=1}^n \log P_i}. \quad (48)$$

The resulting matrices will in most cases be sparse. Phrase and different variations (plural/singular and cases) *система електронної контент-комерції* [systema elektronnoyi kontent-komertsiyi] (electronic content commerce system):

$$\begin{aligned} P(\text{систем електрон контент комерц}) &= \\ &= P(\text{електрон}|\text{систем})P(\text{контент}|\text{електрон})P(\text{комерц}|\text{контент}) = \\ &= 0,124 \times 0,81 \times 0,179 = 0,01797876. \end{aligned}$$

## 5. Conclusions

The general architecture of computer linguistic systems is developed based on the main processes of processing information resources such as integration, maintenance and content management, as well as using methods of intellectual and linguistic analysis of text flow using machine learning technology. The IT of intellectual analysis of the text flow based on the processing of information resources has been improved, which made it possible to adapt the generally typical structure of content integration, management and support modules to solve various NLP problems and increase the efficiency of CLS functioning by 6-9%. This became possible thanks to the combination of linguistic analysis methods adapted to the Ukrainian language, improved IT processing of information resources, ML and a set of metrics for evaluating the effectiveness of CLS functioning. The main principle of building such CLS is modularity, which facilitates their construction according to the requirements for the availability of appropriate processes for solving a specific NLP problem. The main NLP methods based on regular expression matching with patterns in grapheme and morphological analyses of Ukrainian-language texts are described. NLP methods based on pattern-matching regular expressions have been improved, which made it possible to adapt methods of text tokenization and normalization by cascades of simple substitutions of regular expressions and finite state machines. The main valid operations of regular expressions are defined as union and disjunction of symbols/strings/expressions, number and precedence operators, as well as anchors as special symbols for identifying the presence/absence of symbols in RE. The main stages of tokenization and normalization of the Ukrainian text by cascades of simple substitutions of regular expressions and finite state machines are defined. The MA method of the Ukrainian-language text based on word segmentation and normalization, sentence segmentation and modified Porter's stemming algorithm was improved as an effective means of identifying lem affixes for the possibility of marking the analysed word, which made it possible to increase the accuracy of keyword searches by 9%. Algorithms for word segmentation and normalization, sentence segmentation, and Porter's modified stemming are implemented and described as an effective way of identifying lem affixes for the possibility of marking the analysed word. Unlike the classic Porter algorithm (it does not have high accuracy even for English-

language texts), the modified one is adapted specifically for the Ukrainian language and gives an accurate result in 85-93% of cases, depending on the quality, style, genre of the text and, accordingly, the content of CLS dictionaries. The algorithm for the minimum editorial distance of lines of Ukrainian texts is described as the minimum number of operations necessary to transform one into another.

## References

- [1] B. Bengfort, R. Bilbro, T. Ojeda, Applied text analysis with Python: Enabling language-aware data products with machine learning. O'Reilly Media, Inc. (2018).
- [2] D. Jurafsky, J. H. Martin, Deep Learning Architectures for Sequence Processing. URL: <https://web.stanford.edu/~jurafsky/slp3/9.pdf>.
- [3] D. Jurafsky, J. H. Martin, Naive Bayes and Sentiment Classification. URL: <https://web.stanford.edu/~jurafsky/slp3/4.pdf>.
- [4] D. Jurafsky, Logistic Regression. URL: <https://web.stanford.edu/~jurafsky/slp3/5.pdf>.
- [5] D. Jurafsky, J. H. Martin, Neural Networks and Neural Language Models. <https://web.stanford.edu/~jurafsky/slp3/7.pdf>.
- [6] V. Vysotska, Modern State and Prospects of Information Technologies Development for Natural Language Content Processing, CEUR Workshop Proceedings 3368 (2024) 198-234.
- [7] A. Berko, Y. Matseliukh, Y. Ivaniv, L. Chyrun, V. Schuchmann, The text classification based on Big Data analysis for keyword definition using stemming, in: Proceedings of the IEEE 16th International conference on computer science and information technologies, CSIT-2021, Lviv, Ukraine, 22–25 September 2021, pp. 184–188.
- [8] N. Shakhovska, I. Shvorob, The method for detecting plagiarism in a collection of documents, in: Proceedings of the International Conference on Computer Sciences and Information Technologies, CSIT, 2015, pp. 142-145.
- [9] R. Romanchuk, V. Vysotska, V. Andrunyk, L. Chyrun, S. Chyrun, O. Brodyak, Intellectual Analysis System Project for Ukrainian-language Artistic Works to Determine the Text Authorship Attribution Probability, in: Proceedings of the 18th IEEE International Conference on Computer Science and Information Technologies, CSIT 2023, Lviv, Ukraine, October 19-21, 2023. IEEE 2023.
- [10] V. Lytvyn, P. Pukach, V. Vysotska, M. Vovk, N. Kholodna, Identification and Correction of Grammatical Errors in Ukrainian Texts Based on Machine Learning Technology. Mathematics 11 (4) (2023) 904. <https://doi.org/10.3390/math11040904>
- [11] K. Shakhovska, et al., An approach for a next-word prediction for Ukrainian language. Wireless Communications and Mobile Computing 2021 (2021) 1-9.
- [12] S. Kubinska, R. Holoshchuk, S. Holoshchuk, L. Chyrun, Ukrainian Language Chatbot for Sentiment Analysis and User Interests Recognition based on Data Mining, CEUR Workshop Proceedings 3171 (2022) 315-327.
- [13] T. N. Shakhovska, O. Basystiuk, K. Shakhovska, Development of the Speech-to-Text Chatbot Interface Based on Google API, CEUR Workshop Proceedings 2386 (2019) 212-221.



- [14] V. Husak, O. Lozynska, I. Karpov, I. Peleshchak, S. Chyrun, A. Vysotskyi, Information System for Recommendation List Formation of Clothes Style Image Selection According to User's Needs Based on NLP and Chatbots, CEUR workshop proceedings 2604 (2020) 788-818.
- [15] V. Vysotska, Ukrainian Participles Formation by the Generative Grammars Use, CEUR workshop proceedings 2604 (2020) 407-427.
- [16] V. Vysotska, S. Holoshchuk, R. Holoshchuk, A comparative analysis for English and Ukrainian texts processing based on semantics and syntax approach, CEUR Workshop Proceedings 2870 (2021) 311-356.
- [17] O. Bisikalo, O. Boivan, N. Khairova, O. Kovtun, V. Kovtun, Precision automated phonetic analysis of speech signals for information technology of text-dependent authentication of a person by voice, CEUR Workshop Proceedings 2853 (2021) 276-288.
- [18] O. Bisikalo, V. Vysotska, Linguistic analysis method of Ukrainian commercial textual content for data mining, CEUR Workshop Proceedings 2608 (2020) 224-244.
- [19] I. Khomytska, I. Bazylevych, V. Teslyuk, I. Karamysheva, The chi-square test and data clustering combined for author identification, in: Proceedings of the IEEE XVIIIth Scientific and Technical Conference on Computer Science and Information Technologies, CSIT 2023, Lviv, Ukraine, 19-21 October 2023.
- [20] I. Khomytska, V. Teslyuk, The Multifactor Method Applied for Authorship Attribution on the Phonological Level, CEUR workshop proceedings 2604 (2020) 189-198.
- [21] I. Khomytska, V. Teslyuk, A. Holovatyy, O. Morushko, Development of methods, models, and means for the author attribution of a text, Eastern-European Journal of Enterprise Technologies. 3(2(93)) (2018) 41-46. doi: 10.15587/1729-4061.2018.132052.