ATLANTIS PRESS
Advances in Computer Science Research (ACSR), volume 72
IV International Research Conference "Information Technologies in Science, Management, Social Sphere and Medicine" (ITSMSSM 2017)

A Study of Efficiency of Modern Inflection and Lemmatization Software

Oleg Sychev, Vladislav Gurtovoy, Nikita Penskoy
Software Engineering Department
Volgograd State Technical University
Volgograd, Russia
Email: oasychev@gmail.com

***Abstract* — Inflectors – the programs that generate different forms of the given lemma – are important part of natural language processing, especially natural language generation. They, along with lemmatizers – the programs that determine the lemma of the word – also can be used in teaching grammar. However, writing an efficient inflector is not an easy task and many modern applications have deficiencies. The article compares several modern open source inflectors using same-lemma detection method, which defines if inflector could be used to identify, that two different words belong to the same lemma.**

Keywords — inflection; lemmatization; natural language processing

I. INTRODUCTION

A word change (an inflection) is a word-forming mechanism for expressing various grammatical categories, such as tension, mood, voice, aspect, person, gender, number and case. The fusional morphology is often implemented by concatenating related morphemes (prefixes and suffixes) with a root, but modifications, such as consonant duplicating and vowel change are also used [1].

Automated word inflection is one of the problems of natural language processing. The most popular application field for word inflection is various online stores, where nouns are inflected for the number when the quantity of the goods is changed. In some programming languages, such as Ruby, noun inflection is used to generate a method name, automatically generating the plural form of nouns. A special case of the inflection problem for verbs occurs in the generation and analysis of texts. Inflection helps to compare verbs in different forms, though methods of lemmatization and stemming also can be used for this purpose [2]. The problem of word-change also comes up in the corpus linguistics, where it can be used together with lemmatization in order to increase the accuracy of classification [3]. The task of word inflection also comes up when learning a foreign language, in order to understand the rules of word formation in a specific case using an example.

Word inflection is a difficult task because of the large quantity of rules and exceptions. It is solved by various approaches, with each approach imposing additional restrictions on a set of words that will be correctly inflected.

II. STATE OF THE ART

Most word inflection applications do noun inflections, allowing to form plural and singular forms. These tools cover a set of tasks that often come up when running various online services. In most cases of noun inflection, a fixed set of rules is used for altering words while application maintains a database of exceptions.

Generalizations of this approach for several parts of speech are also used. The typical example is en-inflectors library which is written on the basis of Node.js platform and has a sufficiently large volume of tests [4].

The natural approach for solving the complete problem of word inflection is to create a dictionary, which contains words, their respective lemmas and parts of speech [5]. This approach requires a large quantity of resources to create an extensive database of words. However, it provides greater accuracy of the conversion compared to other approaches. The example of this approach is SimpleNLG library for the English language which is written in the Java programming language [6].

Another possible approach is combining the compilation of a dictionary and predicting the word form if the necessary one was not found in the dictionary. This form is based on the data about similar words. A popular tool for Russian and English languages which uses this approach is the phpMorphy library which is written in the PHP programming language [7].

Yet another approach is to make up a set of rules for words that use common rules to construct different forms [8]. This set of rules can be made up manually or can be the result of machine learning on a large set of words and their declensions. The results obtained through machine learning show good accuracy and can be extended for use in different languages, but they still have less accuracy compared with rule-based and dictionary-based approaches [9].

III. METHODS AND SAMPLE

This paper compares the accuracy of several tools for word inflection in the English language that were selected by the authors and a rule-based tool written by the authors (word_operators_inflector) to determine the efficiency of different approaches. The tools for declining nouns, adjectives

and adverbs, and conjugating verbs were chosen for complex evaluation during the experiment.

Authors used two approaches to evaluate the efficiency of inflection software. Lemmatization approach was used besides usual conversions between different forms of the word. It was assumed that inflector which can be used to correctly determine whether two words are the forms of one lemma will inflect both words correctly. This hypothesis can be verified by finding the number of words on which the tool has successfully coped with the task of word change and has not coped with the task of lemmatization and the number of words on which the tool has successfully coped with the task of lemmatization and has failed to do the task of word inflection.

The proposed comparison scheme consists of changing the word in a given form to a lemma and comparing it with the assumed lemma, then changing the word from the lemma to a given form. Some lemmatizers, that don't perform word inflection, were included in the list of compared tools in order to compare their efficiency in lemmatization with word inflection tools.

English is considered to be weakly inflected language. Inflections are divided into 2 groups: declining and conjugating. Conjugation refers to verbs that change their form to express different grammar categories. Declension refers to other parts of speech that can undergo inflection. They are nouns, pronouns, adjectives and adverbs [10].

Verbs are conjugated to reflect tense, aspect, mood, voice, person and speech. All these categories are formed using auxiliary verbs and verb participles [11].

There are technically only two grammatical verb tenses in English: the past tense and the present tense. Verbs in their basic form inherently describe the present time, and they can be conjugated into a unique form that describes the past. English verbs have no future tense in the strict sense because there is no unique verb form to denote future action. Nevertheless, we commonly refer to several structures that are used for future meaning as belonging to the future tense. While tense is shows when an action, state of being, or event occurs, grammatical aspect is concerned with how it occurs in time. Each verb tense has four aspects, or temporal structures: the simple, the perfect, the continuous, and the perfect continuous (see table 1).

TABLE I. TEMPORAL STRUCTURES OF THE VERB TENSE

Tense	Aspect	Structure
present	simple	subject + infinitive verb (VBP)
	perfect	subject + have/has + past participle (VBD)
	continuous	subject + is/are + present participle (VBG)
	perfect continuous	subject + have/has + been + present participle (VBG)
past	simple	subject + past verb (VBN)
	perfect	subject + had + past participle(VBD)
	continuous	subject + was/were + present participle (VBG)
	perfect continuous	subject + had + been + present participle (VBG)
future	simple	subject + will + infinitive verb (VBP)
	perfect	subject + will have + past participle(VBD)
	continuous	subject + will+ be + present participle (VBG)
	perfect continuous	subject + will + have + been + present participle (VBG)

As we can see from the table 1, the unique verb forms are the infinitive form (VBP), past form (VBD), past participle form (VBN), present participle form (VBG).These forms should be used in the experiment.

The vast majority of verbs add suffix '-d' or '-ed' to form VBD and VBN. But there are many verbs that form VBD and VBN by adding other suffixes, prefixes and infixes and by modifications, such as consonant duplicating and vowel change. VBG is formed by adding suffix '-ing' to the end of the verb.

Grammatical person refers to the degree of involvement of a participant in an action, event, or circumstance. There are three degrees of grammatical person: first person (the speaker), second person (someone being spoken to), and third person (anyone/anything not being directly addressed).

The vast majority of verbs only conjugate for the third-person singular subjects (he, she, and it) by taking suffix '-s' or '-es'. However, the verb "be" is unique. It has five different conjugations according to the grammatical person of its subject and the tense of the verb. The authors decided to add only the third person present form (VBZ) of the verb to the experiment because the only verb "be" has more than two forms according to the grammatical person of its subject and the tense of the verb.

Grammatical mood refers to the way in which a verb is used to express certain meaning by the speaker. Grammatical voice describes the relationship between the verb and the subject in a sentence. Every kind of grammatical mood (the indicative mood, the subjunctive mood and the imperative mood) and verb constructions of active voice, passive voice and middle voice use VBP, VBD, VBN, VBG or VBZ in their verb construction, so covering them doesn't require adding new verb forms.

Grammatical speech refers to how we report what another person said, which affects the conjugation of the verbs that are used. Direct speech refers to the direct quotation of something that someone else said. Because the quote happened in the past, the reporting verb is put into VBD, but the verbs used within the quotation don't change (VBP, VBD, VBN, VBG and VBZ can be used there). Reported or indirect speech is used when it is necessary to paraphrase what somebody said, rather than directly quoting him. In reported speech, according to conventional rules, we must change the paraphrased information one degree into the past as shown in table 2. So, the inflection of the verb can be tested by testing conversion between each possible pair of five main forms of the verb: VBP, VBD, VBN, VBG and VBZ.

TABLE II. TENSE SHIFT IN REPORTED SPEECH

Direct speech	Reported speech
I live (VBP) in France.	He said he lived (VBN) in France.
He is (VBZ) washing (VBG) the dishes.	He told (VBN) us he was (VBN) washing (VBG) the dishes.
I was (VBN) a basketball player.	He said (VBN) that he had (VBD) been (VBD) a basketball player.
She was (VBN) cooking (VBG) when you came.	She told (VBN) me you had (VBD) been (VBD) cooking (VBG) when I came (VBN).

Nouns are declined for the number [11]. Plurals of nouns are used to indicate when there is more than one person, place, animal, or thing. Nouns form the plural by adding suffix '-s' or '-es' to the end of the word. Nouns are also declined to reflect gender (as in prince (masculine) vs. princess (feminine)). Since most nouns don't change when inflected for gender, only pluralization abilities should be tested in the experiment. So, tests for noun inflection include comparing infinitive form (NN) with plural form (NNS).

Very little number of pronouns is a subject of inflection. Only the pronoun "who", whose object form is "whom" and possessive is "whose", and personal pronouns are declined. There are only 8 personal pronouns ("I", "you", "he", "she", "it", "we", "you", "they"). They are declined based on the grammatical number, person, gender, and case. As we can see, the quantity of inflected pronouns is minor, so the personal pronouns' declension is not examined in the experiment.

The rules for adjectives and adverbs inflection to create noninfinitive forms are equal [11]. Adjectives, as adverbs, add suffix '-er' to form comparative form and suffix '-est' to form superlative form. So in the experiment there were used the infinitive (RB), comparative (RBR) and superlative (RBS) forms of adverbs and infinitive (JJ), comparative (JJR) and superlative (JJS) forms adjectives.

All inflection forms of noun, verb, adjective and adverb that were selected for the experiment are shown in table 3. Tests were divided into 3 main categories: tests for noun inflection, tests for adjective and adverb inflection and tests for verb inflection.

The tests for noun inflection contained nouns, that end in consonant + suffix '-y' (drops suffix '-y' and adds suffix '-ies' pluralizing), suffix '-ie' (suffix '-ies' turns to suffix '-ie' lemmatizing, not suffix '-y'), vowel + suffix '-y' (adds suffix '-s' pluralizing), suffixes '-ss', '-x', '-sh', '-ch', '-z' (adds suffix '-es' pluralizing), vowel + suffix '-o' (adds suffix '-s' pluralizing), consonant + suffix '-o' (adds suffix '-es' pluralizing), foreign word and musical terms ending in suffix '-o' (adds suffix '-s' pluralizing), suffixes '-oe' (adds '-s' pluralizing), '-f', '-fe' (suffixes '-f', '-fe' turn to '-ves' pluralizing), including irregular suffixes '-f', '-fe' nouns (adds suffix '-s' pluralizing), latin nouns (have their own rules of pluralization) and irregular nouns (defy spelling guidelines).

The tests for adjectives and adverbs contained monosyllabic adjectives, that end in suffix '-e' (adds suffix '-r' in comparative and suffix '-st' in superlative), single vowel + consonant, suffix '-y' (drops suffix '-y' and adds suffix '-ier' in comparative and suffix '-iest' in superlative), other monosyllabic adjectives (adds suffix '-er' in comparative and suffix '-est' in superlative), disyllabic adjectives ending in suffix '-y', suffix '-ow', suffix '-er' (uses same rules of declension as monosyllabic adjectives), adjectives, that use both methods of declension (suffix '-er', suffix '-est' and prepending words "more", "most"), polysyllabic adjectives (are prepended by "more" in comparative and "most" in superlative) and irregular adjectives (defy spelling guidelines).

The tests for verbs contained irregular verbs (defy spelling guidelines), compound irregular verbs (inflected like irregular verbs), regular verbs (add suffix '-ed' in past and past participle forms), ending in suffix '-e' (add suffix '-d' in past and past participle forms), suffix '-y' (drop suffix '-y' and add suffix '-ied' in past and past participle forms), stressed last vowel + consonant (double last consonant and add suffix '-ed' in past and past participle forms), not stressed last vowel + consonant (don't double last consonant and add suffix '-ed' in past and past participle forms), consonant 'l', that is doubled while conjugation, consonant 'l', that is not doubled while conjugation, consonants '-ch', '-ss', '-sh', '-x', '-z', '-o', '-c' (add suffix '-ed' in past and past participle forms), doubled consonant (don't drop the last consonant while lemmatization).

TABLE III. INFLECTION FORMS, SELECTED FOR EXPERIMENT

Attribute	Tense	Number	Degree
noun	-	single	-
noun	-	plural	-
verb	infinitive	single	-
verb	third person present	single	-
verb	past	single	-
verb	past participle	single	-
verb	present participle	single	-
adjective/adverb	-	-	infinitive
adjective/adverb	-	-	comparative
adjective/adverb	-	-	superlative

To form testing sample English dictionary from the LanguageTool project [12] was used. The dictionary contained about 350 000 entries. Each entry consists of a lemma, word form and word form tags from Penn Treebank tagset.Quantity of pairs "lemma-word" for each form that was used in the experiment is shown in table 4. The distribution of words was uneven, but since results for each word form was calculated independently it hadn't affected the final result. The distribution of words in the experiment is close to the distribution of words in dictionaries of the English language [13]. The number of words in each category is large enough to draw statistically correct conclusions for the whole category.

To find out whether tested software can be safely used to determine whether two words belonging to the same lemma, the words from the dictionary were paired, the number of relevant pairs used in the experiment is shown in table 5.

IV. DISCUSSION

For each tool and for each pair of words two results were determined. First, the programming tool was used to determine whether the pair belongs to the certain lemma, this can be done for both inflection tools and lemmatizers. Since all pairs were constructed from one lemma, a positive result shows that the tool worked correctly in this case, thus a portion of the false negative results can be determined for each tool.

The second result is calculated only for tools that have the ability to inflect a given lemma. This result shows the correctness of word changing to both word forms. The second result is used to test the hypothesis of the validity of comparing inflection applications based on their lemmatization ability.

Table 6 shows the average values of the results of reduction to one lemma (first result), grouped by inflection tools and type of transformation. In table 7 you can see the average values of the results of inflection from lemma to the given form.

TABLE IV. DISTRIBUTION OF WORDS BY WORD FORMS

Word form	NN	NNS	JJ	JJR	JJS	VBP	VBG	VBZ	VBD	VBN
amount	78876	73235	55445	2480	2133	8581	8535	8402	8642	8676

TABLE V. WORD PAIR DISTRIBUTION, USED IN TESTING

Initial pairs	JJ-JJR	JJ-JJS	JJR-JJS	NN-NNS	VBP-VBD	VBP-VBN	VBP-VBZ	VBP-VBG	VBD-VBN	VBD-VBZ	VBD-VBG	VBN-VBZ	VBN-VBG	VBZ-VBG
amount	2331	1989	2110	54827	8328	8327	8350	8324	8348	8294	8300	8293	8299	8283

TABLE VI. THE AVERAGE VALUES OF THE RESULTS OF REDUCTION TO ONE LEMMA, GROUPED BY MEANS AND TYPE OF TRANSFORMATION

Tool	NLTK_lemmatizer	phpMorphy_inflector	SimpleNLG_inflector	ruby_lemmatizer	skyeng_lemmatizer	word_operators_inflector
JJ/JJR	0.496353	0.719434	0.141141	0.159588	0.676963	0.767053
JJ/JJS	0.540473	0.820010	0.148819	0.162896	0.706385	0.778783
JJR/JJS	0.570616	0.881517	0.152607	0.152133	0.769194	1.000000
NN/NNS	0.447371	0.926551	0.037354	0.153756	0.451475	0.928174
VBP/VBD	0.875120	0.976585	0.147214	0.165706	0.897935	0.544669
VBP/VBN	0.875105	0.977063	0.146511	0.170650	0.896962	0.544254
VBP/VBZ	0.894371	0.989341	0.148982	0.169102	0.904431	0.931257
VBP/VBG	0.882749	0.976454	0.148967	0.171672	0.906896	0.550457
VBD/VBN	0.994370	0.997365	0.984547	0.166267	0.995448	0.994849
VBD/VBZ	0.885580	0.975886	0.143477	0.168435	0.888835	0.604775
VBD/VBG	0.886988	0.976867	0.143494	0.165663	0.896145	0.969880
VBN/VBZ	0.885807	0.976124	0.142892	0.167008	0.888219	0.604365
VBN/VBG	0.887336	0.977347	0.143150	0.177612	0.895650	0.970960
VBZ/VBG	0.895569	0.974647	0.147652	0.166244	0.902692	0.609079

Table 6 shows that in doing lemmatizing the best software is inflector phpMorphy. It is strictly better than other tools with notable exception of word_operators_inflector (a lemmatizing software written by the authors) for the adjectives. It should be noted that lemmatization is included in the list of official functions for phpMorphy.

TABLE VII. AVERAGE VALUES OF THE RESULTS OF INFLECTION FROM LEMMA TO THE GIVEN FORM.

Tool	phpMorphy_inflector	SimpleNLG_inflector
JJ/JJR	0.606607	0.906049
JJ/JJS	0.664153	0.919558
JJR/JJS	0.624645	0.864929
NN/NNS	0.815128	0.942656
VBP/VBD	0.899015	0.885927
VBP/VBN	0.895040	0.882431
VBP/VBZ	0.927066	0.984192
VBP/VBG	0.916266	0.914704
VBD/VBN	0.891830	0.879612
VBD/VBZ	0.894381	0.878466
VBD/VBG	0.897590	0.883494
VBN/VBZ	0.890269	0.875075
VBN/VBG	0.893361	0.880106
VBZ/VBG	0.912954	0.907280

Another inflection tool – SimpleNLG – shows rather poor results. SimpleNLG is a library which primary aim is text generation and it works better inflecting lemmas to different word forms than between lemmatizing word forms. In table 7 it can be seen that SimpleNLG shows a better results than phpMorphy on tests with adjectives and nouns, and worse results on tests with verbs.

Relative error of the lemmatizing and inflection is shown in table 8 (inf_fail is a portion of cases where lemmatization worked correctly, but inflection did not; lem_fail is a portion of cases where inflection worked correctly, but lemmatization did not). It can be seen that phpMorhpy consistently performs better at lemmatization while SimpleNLG performs better at inflection.

SimpleNLG shows much better results with inflection than with lemmatization. Tables 6 and 7 show that each tool has failed tests at least in one of the categories, so we can calculate the general coverage of tests in which at least one tool shows a positive result. This will show maximum level of efficiency that potentially could be achieved by using mentioned tools together (see table 9). According to the table 9, the total accuracy of both lemmatization and inflection is quite high. If a way of combining different tools with the guaranteed selection of the correct result can be found, the accuracy of such lemmatizer will be 92% for the worst case among adjectives and close to the 100% among nouns and verbs.

TABLE VIII. RELATIVE ERROR FOF THE LEMMATIZING AND INFLECTION FOR THE TOOLS THAT PERFORM BOTH OPERATIONS

type\tool	phpMorphy_inflector		SimpleNLG_inflector	
	inf_fail	*lem_fail*	*inf_fail*	*lem_fail*
JJ/JJR	0.115830	0.003003	0.000429	0.765337
JJ/JJS	0.167924	0.012066	0.000000	0.770739
JJR/JJS	0.268720	0.011848	0.001422	0.713744
NN/NNS	0.126452	0.015029	0.011545	0.916848
VBP/VBD	0.079491	0.001921	0.004563	0.743276
VBP/VBN	0.083944	0.001921	0.004563	0.740483
VBP/VBZ	0.067665	0.005389	0.000719	0.835928
VBP/VBG	0.062110	0.001922	0.000120	0.765858
VBD/VBN	0.105534	0.000000	0.104935	0.000000
VBD/VBZ	0.086810	0.005305	0.001085	0.736074
VBD/VBG	0.081566	0.002289	0.000723	0.740723
VBN/VBZ	0.091161	0.005306	0.001206	0.733390
VBN/VBG	0.086275	0.002289	0.000964	0.737920
VBZ/VBG	0.068212	0.006519	0.001087	0.760715

Table 10 contains data about possible improvement of phpMorphy by combining it with other tools in cases when they perform better. It can be seen that phpMorphy lemmatizing efficiency can be significantly improved in lemmatizing adjectives (where it performs poorly). Improvement to other parts of speech is minor because phpMorphy already performs very well for them.

TABLE IX. TOTAL COVERAGE OF TESTS BY INFLECTION TOOLS

Type	Result	Revert
JJ/JJR	0.920206	0.939511
JJ/JJS	0.922071	0.950729
JJR/JJS	1.000000	0.912606
NN/NNS	0.987343	0.970363
VBD/VBG	0.996732	0.954848
VBD/VBN	0.999157	0.950157
VBD/VBZ	0.995535	0.953415
VBN/VBG	0.997942	0.950847
VBN/VBZ	0.995534	0.949547
VBP/VBD	0.994837	0.955337
VBP/VBG	0.996396	0.969249
VBP/VBN	0.994957	0.951489
VBP/VBZ	0.998803	0.994013
VBZ/VBG	0.996858	0.968338

TABLE X. LEMMATIZING EFFICIENCY IMPROVEMENT WHEN COMBINING THE RESULTS OF PHPMORPHY WITH OTHER TOOLS

type\tool	phpMorphy_inflector
JJ/JJR	0.200772
JJ/JJS	0.102061
JJR/JJS	0.118483
NN/NNS	0.060792
VBD/VBG	0.019864
VBD/VBN	0.001793
VBD/VBZ	0.019648
VBN/VBG	0.020595
VBN/VBZ	0.019410
VBP/VBD	0.018252
VBP/VBG	0.019943
VBP/VBN	0.017894
VBP/VBZ	0.009461
VBZ/VBG	0.022211

V. CONCLUSION

So results of research shows that the best available tool for the English language inflection is SimpleNLG while lemmatizing is performed better with phpMorphy. PhpMorphy is very good at lemmatizing most parts of speech except adjectives, but its performance for adjectives can be significantly improved by combining with other tools.

REFERENCES

[1] J. Hajič and B. Hladká “Tagging inflective languages: Prediction of morphological categories for a rich, structured tagset” Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics, vol. 1, Association for Computational Linguistics, 1998, pp. 483-490.

[2] L. Lv and S.Y. Liu “Research of English text classification methods based on semantic meaning” Information and Communications Technology, 2005, Enabling Technologies for the New Knowledge Society: ITI 3rd International Conference on IEEE, 2005, pp. 689-700.

[3] M. F. B. do Nascimento, L. Pereira and J. Saramago “Portuguese corpora at CLUL” PRAXIS, 2000, V. 2. №. 2.1/759, p. 95.

[4] English Inflectors Library, https://www.npmjs.com/package/en-inflectors

[5] J. et al. Hajic “Prague dependency treebank 2.0” CD-ROM, Linguistic Data Consortium, LDC Catalog No.: LDC2006T01, Philadelphia, 2006., T. 98.

[6] A. Gatt and E. Reiter “SimpleNLG: A realisation engine for practical applications” Proceedings of ENLG, 2009.

[7] phpMorphy, http://phpmorphy.sourceforge.net/dokuwiki/

[8] W. D. Foust “Automatic English Inflection” Proceedings of the National Symposium on Machine Translation (Englewood Cliffs, NJ, Prentice-Hall, Inc., 1961), 1960, pp. 229-233.

[9] G. Nicolai, C. Cherry and G. Kondrak “Inflection Generation as Discriminative String Transduction” HLT-NAACL, 2015, pp. 922-931.

[10] “The Farlex Grammar Book”, P.: Farlex International, 2016.

[11] H.H. Schmidt “Advanced Grammar” L.: Pearson PLC, 2015.

[12] M. Miłkowski “Developing an open-source, rule-based proofreading tool” Software: Practice and Experience, 2010, T. 40, №. 7, C. 543-566.

[13] C. Fellbaum “WordNet”, John Wiley & Sons, Inc., 1998.