Received: August 14, 2020. Revised: October 19, 2020.

International Journal of
Intelligent Engineering & Systems

http://www.inass.org/

Automated Identification of Semantic Similarity between Concepts of Textual Business Rules

Abdellatif Haj[1]* **Youssef Balouki[1]** **Taoufiq Gadi[1]**

[1]*Laboratory of Mathematics, Computer and Engineering Sciences, Department of Maths, Faculty of Sciences and Techniques, Hassan I University, Settat, Morocco*
* Corresponding author's Email: haj.abdel@gmail.com

Abstract: Business Rules (BR) are usually written by different stakeholders, which makes them vulnerable to contain different designations for a same concept. Such problem can be the source of a not well orchestrated behaviors. Whereas identification of synonyms is manual or totally neglected in most approaches dealing with natural language Business Rules. In this paper, we present an automated approach to identify semantic similarity between terms in textual BR using Natural Language Processing and knowledge-based algorithm refined using heuristics. Our method is unique in that it also identifies abbreviations/expansions (as a special case of synonym) which is not possible using a dictionary. Then, results are saved in a standard format (SBVR) for reusability purposes. Our approach was applied on more than 160 BR statements divided on three cases with an accuracy between 69% and 87% which suggests it to be an indispensable enhancement for other methods dealing with textual BR.

Keywords: Business rules, Semantic similarity, Synonym extraction, Abbreviation identification, NLP.

1 Introduction

Business Rules are expected to assert the structure or control the behaviour of the business [1]. They should be separated from other layers and presented in an unambiguous language to all stakeholders [2]. The heterogeneity of different intervenors coupled with the divergence of their interests make the natural language the most appropriate format to express business rules. Then, IT people tend to transform them to a more formal languages for analyse and design purposes. Doing such transformation manually is time consuming and error prone, which explain the great interest in the last decade to make it automatic.

Approaches dealing with textual business needs largely rely on Natural Language Processing (NLP), or adopt a Controlled Natural Language (CNL) [3] easily understood by machines.

Existing approaches focus on extracting knowledge (semantic) from text, but ignore other issues leading to inconsistencies, resulting specially from the fact that BR are usually written by different stakeholders. Using more than one designation for the same concept is one of theme.

BRs are generally neither few statements to be manually processed, nor enough to be processed using machine learning methods. Probably, this justifies the fact that approaches tend to ignore the existence of synonyms [4], or provide a user interface (UI) to add theme manually [5], or assume the prior existence of a domain knowledge containing the business vocabulary [6]. While other approaches try to identify synonyms at advanced steps of the development process [7]. However, the identification of semantic similarity between terms at early steps will - without a doubt - improve the quality of next steps models, and then decrease the number of iterations as well as the process time.

In this paper, we present an approach to enhance existing approaches dealing with NL BR statements by automatically identifying synonyms using NLP, knowledge-based algorithm, and heuristics.

Our approach was tested over more than 160 BR statements divided on 3 different cases, with an accuracy between 67% and 80%.

This paper should be of interest to all software engineering people who work with natural language, especially those who are interested in transforming text to formal specifications. This approach makes the following contributions:

1. Enhance methods dealing with Text to Model transformation, by identifying terms with same meaning.
2. Improve business vocabulary formulation.
3. Preserve the BR integrity.

Automatic extraction of synonym terms is largely related to semantic similarity (relatedness) identification between entities, namely: terms, phrases, paragraphs, or documents [8-9]. To identify Word-to-Word similarity, 3 main methods are used in the literature [10] : (i) String-based methods in which similarity between two character strings are measured; (ii) Corpus-based methods use statistics collected from collection of texts (Corpus); and (iii) Knowledge-based methods which depend on predefined linguistic information derived from semantic networks such as WordNet [11]. Certainly, these methods are not all suitable to our objective. For example, string-based methods surely will identify entities having close sequence of characters; but result such as "renter" and "rental" are considered false positive in our case. As well, knowledge-based methods alone are not sufficient, and must be supported by the context in which the term was found, as clearly shown in the definition of the "*synonym*" concept: "*words that denote the same concept and are interchangeable in many contexts*" (WordNet [11]). Regarding corpus-based approaches, the context plays an important role. For example, sentences may be transformed to a Vector Space Models [12] to extract their meaning. Word2Vec method [13] is widely used at this level to establish semantic similarity via word statistics which relay on the context in which terms are used. The accuracy of methods relying on this solution, is related to the large number of data available at the input size.

As a solution, we tried to put ourselves in the shoes of business writers, to recognize the different possible sources of synonym before making the following hypotheses:

H1) Terms of the same statement are considered neighbours of each other. In other words, all terms of the same statement participate somehow to construct the context of each term of this statement.

H2) Terms having no common neighbour means that they have different meaning, even if they are synonym according to dictionaries.

H3) Terms which are neighbours are not synonym. Which means that terms appearing in at least one common statement have two different meaning.

BR statements should be atomic, well-formed, and written in business terms [1] instead of paragraphs. Thus, limiting term context to terms surrounding it (such in machine learning methods) is not a good choice in BR environment (H1 & H2). Furthermore, one BR statement is often formulated by one person; thus, the probability to find synonym terms in the same statement is very low compare to other statements written by other people (H3). Finally, we consider that the writer of statement intentionally chooses proper terms for proper expressions; thus hypernym (hyponym) are not considered as synonym. Altogether, in our word we decided to identify terms having same dictionary-based sense before to be refined using heuristics. In addition, a solution to relate abbreviations with their expansions (definitions) is also proposed.

The next section looks at existing approaches dealing with synonym identification from natural language specifications; Section 3 details our approach to identify synonym terms from textual BR as well as abbreviation-expansion pairs; Section 4 evaluates our method and discuses obtained results; and Section 6 draws the limits coupled with directions for future work and concludes the paper.

2 Related works

Despite the problems that may arise from the existence of more than one designation with same meaning, synonyms identification is neglected in most approaches dealing with NL BRs as can be seen in Table 1.

In this section we will give an overview of existing approaches dealing with synonyms identification from natural language specifications.

In other approaches, there is an important difference in the way semantic similarity between terms is handled. V. Gruhn [15] have created a catalogue of synonyms and antonyms to help in detecting error patterns that can be found in Event-Driven Process Chains. A. Awad [16] proposed an automated approach for querying a business process model repository, in which the most common sense was selected from WordNet [11] dictionary. Other approaches tried to extract synonyms using dictionary (like WordNet) without explanation neither about what was the algorithm used, nor if the context was taken into consideration or not, such in [17, 18] in which SBVR business vocabularies and rules are extracted from UML use case diagrams; or [19] which presented an automatic approach to generate BPMN models from natural language text;

Table 1. A comparative table of approaches dealing with synonyms in textual specifications

Approach	Input	Output	Synonym Identification
P. K. Chittimalli [4]	Textual Business Rules	SBVR: Entities, facts, and rules	No
S. Roychoudhury [5]	legal English text	SBVR model	Manual
I. S. Bajwa [6]	Textual specification	OCL Constraints	Manual
M. Selway [14]	Textual Business specification	Formal model	Manual
V. Gruhn [15]	Event-Driven Process Chains	error patterns	Manual
P. Danenas [17]	Use Case Diagram	SBVR Business Vocabularies and Rules	Based entirely on dictionary results. (method not cited)
T. Skersys [18]	Use Case Diagram	SBVR Business Vocabularies and Rules	Based entirely on dictionary results. (method not cited)
F. Friedrich [19]	Natural Language Text	BPMN models	Based entirely on dictionary results. (method not cited)
Y. Zhang [20]	English requirements	requirement-to-code links	Based entirely on dictionary results. (method not cited)
A. Awad [16]	Business Process model query	List of process models	Based entirely on dictionary result (most common sense)
M. Ehrig [21]	Business Process models	Similarity between Business Process models	Semi-automatic: require a pre-existing ontology-based vocabulary.
F. Pittke [7]	Textual Process models	Homonyms and synonyms terms	Automatic: Combine scores computed from translations in different languages using BabelNet
F. Dalpiaz [24]	user stories	near-synonyms terms	Automatic: calculate the similarity between terms then between their context using Cortical.io
Our Method	Natural Language Business Rules	Synonyms terms & abbreviation/expansion saved according to SBVR standard.	Automatic: use NLP and knowledge-based algorithm refined using heuristics.

or [20] which combined various features to recover requirement-to-code links. M. Ehrig [21] proposed an approach for (semi-) automatic detection of synonyms and homonyms of process element names using a preexisting ontology-based vocabulary get from UML Profile. They combined three similarity measures, syntactic (compare the number of common characters), linguistic (the number of WordNet senses), and structural (places of concepts with their attributes and transitions). F. Pittke [7] proposed a technique to detect and resolve lexical ambiguities in textual process models caused by homonyms and synonyms by combining scores computed from translations in different languages. To find word with similar senses, they used a graph-based multilingual joint approach based on the multilingual knowledge base BabelNet [22] with XMeans clustering [23]. F. Dalpiaz [24] try to detect terminological ambiguity in user stories by calculating the similarity between terms then between their context. For this reason, Cortical.io [25] is used, which is considered as a very powerful AI-based tool, with a high processing speed. Cortical.io relies on matrices created from a large collection of websites to calculate the semantic similarity.

It can be concluded that, there is no single agreed method to extract synonyms from text which is valid for all goals; but we are facing a non-trivial task that differs depending on the objective and context.

In the end, despite the divergence in their area of application, the last approach [24] remains the closest to ours, as it extracts nouns automatically from a set of sentences, before calculating the similarity by taking the context into consideration. While our approach deals with textual BR statements having no standard format and uses heuristics to improve the accuracy of identified synonyms. Our approach is unique in that it also identifies abbreviations/expansions (as a special case of synonym) which is not possible using a dictionary, before saving obtained results in a standard format (SBVR) for reusability purposes.

Received: August 14, 2020. Revised: October 19, 2020.

3 Our approach

Our aim is to enhance approaches dealing with BR written in natural language by identifying terms having the same meaning.

Initially, our approach begins with a linguistic analysis using an NLP pipeline to tokenize, tag, and generate the parse tree of each statement. After that, single as well as multi word terms are extracted using algorithm 1. Next, synonym terms are identified using algorithm 2. Then, Abbreviations are identified using algorithm 3. Finally, an SBVR-based XMI file [26] is generated containing concepts and their synonyms (Fig. 1). SBVR, or "Semantic of Business Vocabulary and Rules" [26] is an Object Management Group (OMG) standard [27] that help the business vocabulary and rules interchanging amongst organizations in totally technology-independent format.

3.1 Step 1: Linguistic analyses

In this step, each BR statement is splitted into tokens. Then, Stanford CorNLP [28] is used to annotate tokens of each statement with a part-of-speech (POS) tag, (e.g. noun, verb, adjective etc.). Finally, a dependency parse tree is generated for each statement. Fig. 2 shows an example of obtained result.

The output result is saved in an XML file and used as input in the next steps.

3.2 Step 2: Terms extraction

First, for each BR statement, we extract all nouns and mark them as a single word noun. Next, each noun founded in a compound relation with other nouns or having modifiers will be rectified to create a multi word noun and added as a new entry according to algorithm 1.

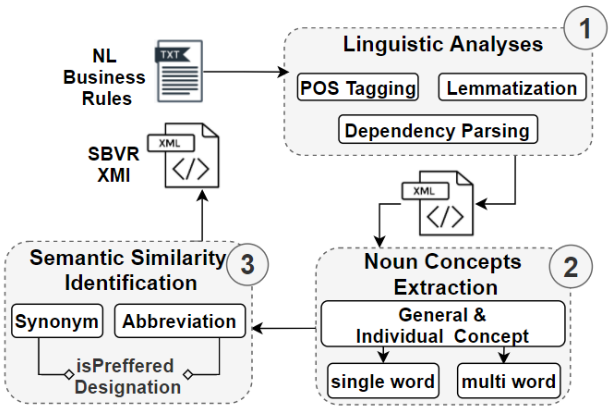


Figure. 1 An overview of our approach

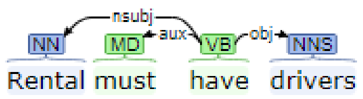


Figure. 2 Example of our NLP pipeline output

Algorithm 1: Noun Concepts Extraction

```
Data: S = all BR statements
Result: NC = all identified Noun Concepts
begin
    for each statement ∈ S do
        for each word ∈ wordsOf(statement) do
            if ¬isNoun(word) then
                skip
            end
            n ← newNounConcept(word)
            setType(n, singleWord)
            NC.add(n)
            // Multiword noun checking
            for each word2 ∈ wordsOf(statement) do
                if isCompoundOf(word2, word)
                  ∨ isModifierOf(word2, word)) then
                    n ← concatenate(word2, n)
                    setType(n, multiWord)
                end
            end
            if getType(n) = multiWord then
                NC.add(n)
            end
        end
    end
    return NC
end
```

3.3 Step 3: Similarity identification

In this step, extracted terms are compared with each other to identify those having same meaning. At the same time, terms having abbreviation format are identified before seeking their expansions (definitions) if exist. Finally, the frequency of each term is calculated to distinguish the preferred designation.

3.3.1. Synonym identification

To label terms as candidate to be semantically similar, they should have a distance greater than "0,5" regarding the dictionary senses. In addition, according to previously cited hypotheses (H1, H2, H3), similar concepts should have at least one common neighbor, but no common statement (see algorithm 2). The distance "0,5" was adopted after a set of experiments (see subsection 4.1).

3.3.2. Abbreviation-expansion identification

In this sub section we will relate terms having an abbreviation format to their expansion if exist.

There is Three main approaches to identify Abbreviation - Expansion pairs [29]: (i) heuristics

Algorithm 2: Synonym identification

```
Data: t1, t2 $\in$ Terms / t1 $\neq$ t2
Result: boolean : t1 and t2 are synonym or not
begin
    if $t1 \in ProperNoun \vee t2 \in ProperNoun$ then
        return false
    end
    //∃t3 ∈Terms ∧ ∃s1,s2 ∈statements/
    //{t1,t3}⊂s1 ∧ {t2,t3}⊂s2 ∧ s1≠s2
    Let Nt1 be terms coexisting with t1 in at least 1 statement
    Let Nt2 be terms coexisting with t2 in at least 1 statement
    if $Nt1 \cap Nt2 = \emptyset$ then
        return false
    end
    // ¬∃s ∈ statements / {t1,t2} ⊂ s
    Let St1 be statements containing term t1
    Let St2 be statements containing term t2
    if $St1 \cap St2 \neq \emptyset$ then
        return false
    end
    if $similarity(t1,t2) < 0.5$ then
        return false
    end
    return true
end
```

approaches (NLP and pattern-based) in which rules are applied [30-35]; (ii) machine learning based approaches [36] in which labelled examples are used for machine training. (iii) Web-Based approaches, in which abbreviation/definitions are identified based on Web resources [37]. There are also hybrid approaches such in [38] which mixes hand-coded constraints with supervised learning.

Web-based approaches will be a good choice for the well-known abbreviations, but business rules usually use a domain specific terminology in which uncommon abbreviations are used. In the other hand, machine learning based approaches require a lot of data for a good machine training which is not possible in limited number of business rule statements. Since abbreviations usually follow standard format defined in their dictionary-based definition; ("*An abbreviation consisting of the first letters of each word in the name of something, pronounced as a word*" - Cambridge dictionary) we decided to use heuristics each of which can reject any candidate abbreviation.

Strong constraints can reject true positive pairs of abbreviation/expansion, such as : an abbreviation must be in one word between parentheses [39-40], or must be in capital letters [35], or adjacent to parentheses [31], or expansions are always surrounding their abbreviations [41]. For this reason, we relied the following algorithm (algorithm 3) that do not go beyond the dictionary-based definition of abbreviation:

Algorithm 3: Abbreviation/Expansion identification

```
Data: abbrev = abbrevaition condidat (concept)
      exp = expansion condidat (concept)
Result: boolean b = abrev and exp are synonym or not
begin
    //∀ a ∈ abbreviations, |wordsOf(a)|=1
    //∀ e ∈ expansions, |wordsOf(e)|>1
    if $( \neg isSingleWord(abrev) \vee \neg isMultiWord(exp) )$ then
        return false
    end
    //∀ a ∈ abbreviations, a ∈ ProperNouns
    if $abrev \neq ProperNoun$ then
        return false
    end
    //∃t ∈Terms ∧ ∃s1,s2 ∈statements/
    //{t,abbrev}⊂s1 ∧ {t,exp}⊂s2 ∧ s1≠s2
    Let Na be terms coexisting with abbrev in at least 1 statement
    Let Ne be terms coexisting with exp in at least 1 statement
    if $Na \cap Ne = \emptyset$ then
        return false
    end
    /*∀ word ∈ exp, firstCharacterOf(word) ∈ characterOf(abbrev)
    for $each\ w \in wordsOf(exp)$ do
        if $firstCharacterOf(w) \notin charactersOf(abbrev)$ then
            return false
        end
    end
    // ∀ c ∈ charactersOf(abbrev), c ∈ charactersOf(exp)
    if $charactersOf(abbrev) \not\subset charactersOf(exp)$ then
        return false
    end
    /* the sequencing of characters in abbreviation should
    respect the sequencing of those characters in its expansion*/
    Let Ca(i) be the character at position(i) in abbrev.characters
    Let ICa(i) be the list of indexes of Ca(i) in exp.characters
    with i > 0 and i<abbrev.lenght-1
    if $min(ICa(i)) \geq max(ICa(i+1)$ then
        return false
    end
    return true
end
```

Algorithm 3 is based on the following heuristics:

Letters of an abbreviation should match the first letter (or letters) of each word in the full form.

Potential abbreviation will be compared with multi words terms (composed of more than one word)

An abbreviation is not a known (dictionary) word. For this reason, they are generally tagged as proper nouns (NNP).

Some special cases are taken into consideration such as:

"X" can be related to a word staring with "EX" (e.g. "Exchange" of "XML")

Lowercased "s" at the end of the abbreviation could be ignored as it is usually used for plural.

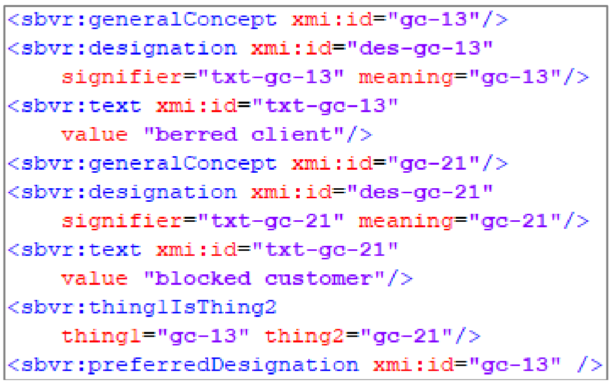
"2" and "4" can mean "to" and "for" respectively.

A candidate abbreviation is compared with its potential expansion in two times: with and without considering prepositions.

3.4 Step 4: SBVR XMI output file

After extracting all terms and identifying synonyms and abbreviations with preferred designations, our result is saved in an XMI file that

Received: August 14, 2020. Revised: October 19, 2020.



```
<sbvr:generalConcept xmi:id="gc-13"/>
<sbvr:designation xmi:id="des-gc-13"
    signifier="txt-gc-13" meaning="gc-13"/>
<sbvr:text xmi:id="txt-gc-13"
    value "berred client"/>
<sbvr:generalConcept xmi:id="gc-21"/>
<sbvr:designation xmi:id="des-gc-21"
    signifier="txt-gc-21" meaning="gc-21"/>
<sbvr:text xmi:id="txt-gc-21"
    value "blocked customer"/>
<sbvr:thing1IsThing2
    thing1="gc-13" thing2="gc-21"/>
<sbvr:preferredDesignation xmi:id="gc-13" />
```

Figure. 3 An excerpt of the SBVR XMI file in which two synonym terms are saved

respect the SBVR XMI Schema [26]. Fig. 3 is an example.

4 Experiments and evaluation

Our experiments were carried out in two steps. First, we have tested some knowledge-based algorithms to select the most accurate one to be adopted in our approach. Then, results obtained from selected algorithm has been refined using our heuristics.

4.1 Selecting knowledge-based algorithm

To choose the most accurate knowledge-based algorithm, we have injected 20 pairs of random synonym terms in a set of 157 BR statements randomly selected from EU-Rent case study found in SBVR documentation [26].

Next, we have used 8 algorithms to identify injected synonyms, namely: WUP [42], RES [43], JCN [44], LIN [45], LCH [46], LESK [47] , HSO [48] and PATH. These algorithms were tested using different distances between terms each time.

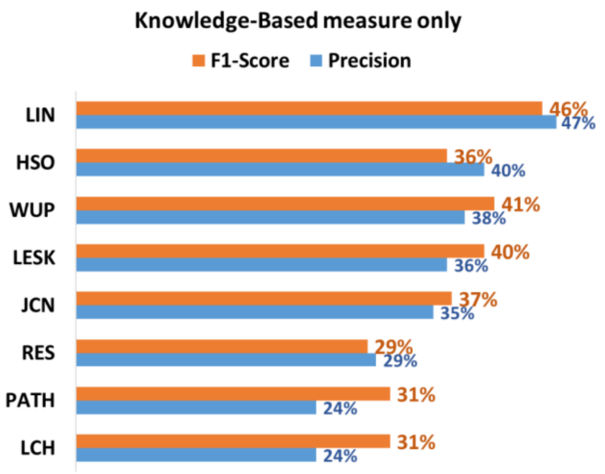


Figure. 4 The precision and F1-score obtained from different knowledge based algorithms

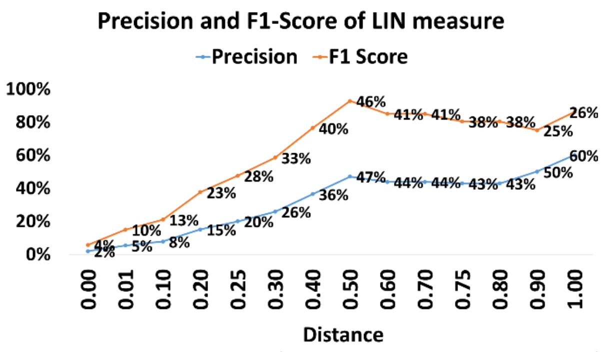


Figure. 5 The accuracy of LIN algorithm by distances for synonyms detection

As can be seen in Fig. 4, LIN algorithm was adopted in our approach as it gives more accurate results compared to others.

Adopted algorithm has been tested using different distances between terms and it has been concluded that by selecting terms having a distance greater than 0,5 we could obtain the most accurate results as can be seen in Fig. 5.

4.2 Validating our approach

Our approach has been tested over more than 160 BR statements taken from 3 different sources: T. Morgan [2] (case 1), G. C. Witt [49] (case 2) and I. Graham [50] (case 3).

To remove all disturbances that can be caused by the ambiguity of the natural language, we have reformulated some statements to follow quality criteria cited in [49]. However, few statements (3%) were incorrectly tagged by Stanford CoreNLP [28]. The source of error at this level was either a confusion between noun and verb (such in “references” and “requests”), or a lack of some punctuation marks which has affected some statement parsing. Nevertheless, 97% of correct analysed statements is considered very satisfying. Table 2 summarises obtained results at this first step of our approach which concerns the linguistic analyse.

To simulate BR writing context, we have divided each of the three cases on 3 parts; then we sent one part of each case to two different engineers and asked

Table. 2 Number of statements and extracted terms for each case

Source	BR statements	Terms	Terms Without repetition	Incorrectly analysed terms
Case 1	48	141	83	2,9%
Case 2	56	216	91	5,5%
Case 3	58	310	66	0,6%

them to reformulate some BR statements by introducing some synonyms and abbreviations. Next, our approach tried to identify injected synonyms and abbreviations.

4.2.1. Synonyms identification

To evaluate synonyms identification, we calculated the number of True Positive (TP), False Positive (FP) and False Negative (FN) leading to calculate the Precision (Eq. (1)), Recall (Eq. (2)) and F1-Score (Eq. (3)).

$$\boldsymbol{Precision} = \frac{TP}{TP + FP} \quad (1)$$

$$\boldsymbol{Recall} = \frac{TP}{TP + FN} \quad (2)$$

$$\boldsymbol{F1 - score} = \frac{2 \times Precision \times Recall}{Precision + Recall} \quad (3)$$

Fig. 6 shows to what extent our heuristics were able to improve the precision of synonyms identified by the adopted knowledge-based algorithm (LIN measure). There was a precision improvement between 29% and 42%, when we have selected synonym terms having at least one common neighbour term.

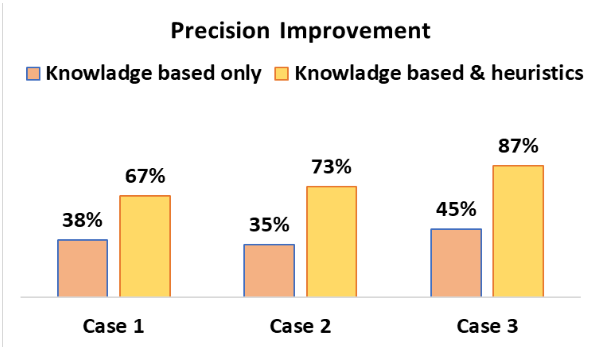


Figure. 6 The precision of synonyms identification using knowledge-based algorithm with and without heuristics

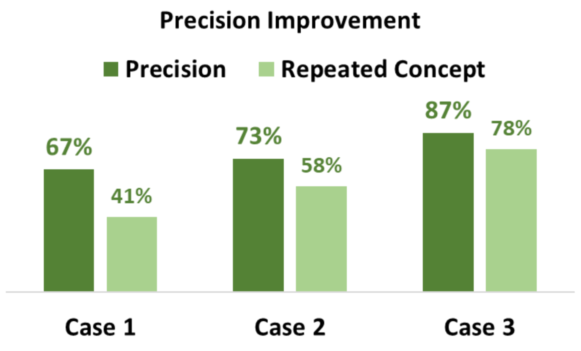


Figure. 7 The relation between the precision of our approach and terms frequency

Table. 3 Our approach results of synonyms identification for each case

Source	Precision	Recall	F1-Score	Repeated terms
Case 1	67%	71%	69%	41%
Case 2	73%	79%	76%	58%
Case 3	87%	87%	87%	78%

The precision, Recall and F1-score for each case are summarized in Table 3 and Fig. 7.

According to Fig. 7, it is clearly recognized that the precision is proportional to the frequency rate of each term. This could be attributed to the fact that more terms appear frequently in different statements, more they reflect a comprehensive meaning, and more their similar terms are likely to be identified. This conclusion is approved by the fact that most synonym pairs that were not identified appear in less than 3 different statements.

Taken together, these results suggest that all terms having at least one common neighbour term but no common statement; and having a distance greater than 0,5 according to LIN algorithm are more likely to be synonym. Nevertheless, few terms that were synonym according to their context and have common terms as neighbours were missed because of their distance less than 0,5 (e.g. “confirmation” and “response” have 0.43). As well, some terms with a distance greater than 0,5 but do not have a common neighbour were also missed.

4.2.2. Abbreviations identification

Regarding abbreviation-expansion, all pairs that have respected the standard format defined by the abbreviation definition cited above, were correctly identified with a high precision. Standard abbreviations such as “second=2nd” and “centimeter=cm” could be easily identified using dictionaries, so they were neglected as they may affect our results.
The precision, Recall and F1-score for each case are summarized in Table 4.
However, few abbreviations have been related to more than one expansion, caused specially by abbreviations composed of two letters such as “BC”

Table 4. Our approach results of abbreviation-expansion identification for each case

Source	Precision	Recall	F1
Case 1	86%	86%	92%
Case 2	86%	86%	92%
Case 3	100%	100%	100%

which is related to both “Business Class” and “Booking Confirmation”. As well, some abbreviations that did not respect the definition were also not identified such as “Average Hours of work per Week = Ah/w”. The latter should be written as “Ahw/w” to fit the definition and then could be identified by our approach.

5 Conclusion

Our contribution is to enhance existing approaches dealing with natural language business rules by automatically identifying synonym terms as well as abbreviation-expansion pairs. Our aim is to extract terms having same meaning but different designations. For this reason, heuristics were used with Natural Language Processing without human intervention. Obtained results are stored in an SBVR-based XMI to facilitate its integration in existing approaches.

Our experiments were carried out on three different groups of business rules from three different domains. Amongst a total of 160 natural language statements, just 3% were incorrectly tagged by the NLP tool. Regarding the accuracy, it ranged between 69% and 87% for synonyms identification, and more than 90% for abbreviation/expansion relations. In the other hand, our heuristics prove their effectiveness with regards to the precision improvement which ranged between +29% and +42%.

The efficiency of our approach relies principally on two factors: (i) Term frequency: as it depends on term’s neighbours to reflect its real meaning. (ii) Knowledge-based measure: which gives the degree of similarity between words using information derived from semantic networks.

The present method has only examined abbreviation-expansion pairs, as well as synonym terms composed of same number of words having a dictionary-based relation. Nevertheless, Other types of synonyms going beyond the scope of this approach should be also identified especially those composed of terms having different senses.

Machine learning methods are undoubtedly the most adequate solution to identify complex terms that have no dictionary-based relation but require rather more data for training which is not usually available in BR context. Thus, we believe that our method could be the most efficient alternative, regarding the relatively small number of BR statements and could be usefully employed to improve approaches dealing with NL BR.

Conflicts of Interest

The authors declare no conflict of interest.

Author Contributions

All authors analysed the data, discussed the results, provided critical feedback, helped shape the research and contributed to the final manuscript.

References

[1] “Business Rules Group”, http://www.businessrulesgroup.org/ (accessed Aug. 02, 2020).

[2] T. Morgan, Business rules and information systems: aligning IT with business goals. Boston: Addison-Wesley, 2002.

[3] T. Kuhn, “A Survey and Classification of Controlled Natural Languages”, *Comput. Linguist.*, Vol. 40, No. 1, pp. 121–170, 2014, doi: 10.1162/COLI_a_00168.

[4] P. K. Chittimalli, C. Prakash, R. Naik, and A. Bhattacharyya, “An Approach to Mine SBVR Vocabularies and Rules from Business Documents”, In: *Proc. of the 13th Innovations in Software Engineering Conf. on Formerly known as India Software Engineering Conf.* Jabalpur, India, pp. 1–11, 2020, doi: 10.1145/3385032.3385046.

[5] S. Roychoudhury, S. Sunkle, D. Kholkar, and V. Kulkarni, “From Natural Language to SBVR Model Authoring Using Structured English for Compliance Checking”, In: *Proc. of 2017 IEEE 21st International Enterprise Distributed Object Computing Conf. (EDOC)*, pp. 73–78, 2017, doi: 10.1109/EDOC.2017.19.

[6] I. S. Bajwa and M. A. Shahzada, “Automated Generation of OCL Constraints: NL based Approach vs Pattern Based Approach”, *Mehran Univ. Res. J. Eng. Technol.*, Vol. 36, No. 2, Art. No. 2, 2017, doi: 10.22581/muet1982.1702.04.

[7] F. Pittke, H. Leopold, and J. Mendling, “Automatic Detection and Resolution of Lexical Ambiguity in Process Models”, *IEEE Trans. Softw. Eng.*, Vol. 41, No. 6, pp. 526–544, 2015, doi: 10.1109/TSE.2015.2396895.

[8] D. R. Kubal and A. V. Nimkar, “A survey on word embedding techniques and semantic similarity for paraphrase identification”, *Int. J. Comput. Syst. Eng.*, Vol. 5, No. 1, p. 36, 2019, doi: 10.1504/IJCSYSE.2019.098417.

[9] M. Farouk and Department of Thermotechnik Computer Science, Assiut University, Markaz El-Fath, Assiut Governorate 71515, Egypt;, “Measuring Sentences Similarity: A Survey”, Indian J. Sci. Technol., Vol. 12, No. 25, pp. 1–11, 2019, doi: 10.17485/ijst/2019/v12i25/143977.

[10] W. H. Gomaa and A. A. Fahmy, “A Survey of Text Similarity Approaches”, *Int. J. Comput. Appl.*, Vol. 68, No. 13, pp. 13–18, 2013.

[11] C. Fellbaum, Ed., WordNet: an electronic lexical database. Cambridge, Mass: MIT Press, 1998.

[12] P. D. Turney and P. Pantel, “From Frequency to Meaning: Vector Space Models of Semantics”, *J. Artif. Intell. Res.*, Vol. 37, pp. 141–188, 2010, doi: 10.1613/jair.2934.

[13] T. Mikolov, K. Chen, G. Corrado, and J. Dean, “Efficient estimation of word representations in vector space”, *ArXiv Prepr. ArXiv13013781*, 2013.

[14] M. Selway, G. Grossmann, W. Mayer, and M. Stumptner, “Formalising natural language specifications using a cognitive linguistic/configuration based approach”, *Inf. Syst.*, Vol. 54, pp. 191–208, 2015, doi: 10.1016/j.is.2015.04.003.

[15] V. Gruhn and R. Laue, “Detecting Common Errors in Event-Driven Process Chains by Label Analysis”, *Enterp. Model. Inf. Syst. Archit. EMISAJ*, Vol. 6, No. 1, Art. No. 1, 2011, doi: 10.18417/emisa.6.1.1.

[16] A. Awad, A. Polyvyanyy, and M. Weske, “Semantic Querying of Business Process Models”, In: *Proc. of 2008 12th International IEEE Enterprise Distributed Object Computing Conf.*, pp. 85–94, 2008, doi: 10.1109/EDOC.2008.11.

[17] P. Danenas, T. Skersys, and R. Butleris, “Natural language processing-enhanced extraction of SBVR business vocabularies and business rules from UML use case diagrams”, *Data Knowl. Eng.*, p. 101822, 2020, doi: 10.1016/j.datak.2020.101822.

[18] T. Skersys, P. Danenas, and R. Butleris, “Extracting SBVR business vocabularies and business rules from UML use case diagrams”, J. Syst. Softw., Vol. 141, pp. 111–130, 2018, doi: 10.1016/j.jss.2018.03.061.

[19] F. Friedrich, J. Mendling, and F. Puhlmann, “Process Model Generation from Natural Language Text”, *in Advanced Information Systems Engineering*, pp. 482–496, 2011, doi: 10.1007/978-3-642-21640-4_36.

[20] Y. Zhang, C. Wan, and B. Jin, “An empirical study on recovering requirement-to-code links”, In: *Proc. of 2016 17th IEEE/ACIS International Conf. on Software Engineering*, Artificial Intelligence, Networking and Parallel/Distributed Computing (SNPD), pp. 121–126, 2016, doi: 10.1109/SNPD.2016.7515889.

[21] M. Ehrig, A. Koschmider, and A. Oberweis, “Measuring similarity between semantic business process models”, In: *Proc. of the Fourth Asia-Pacific Conf. on Comceptual modelling - Volume 67*, Ballarat, Australia, pp. 71–80, 2007, Accessed: Aug. 03, 2020. [Online].

[22] R. Navigli and S. P. Ponzetto, “BabelNet: The automatic construction, evaluation and application of a wide-coverage multilingual semantic network”, *Artif. Intell.*, Vol. 193, pp. 217–250, 2012, doi: 10.1016/j.artint.2012.07.001.

[23] T. Ishioka, “Extended K-means with an Efficient Estimation of the Number of Clusters”, *in Intelligent Data Engineering and Automated Learning — IDEAL 2000. Data Mining, Financial Engineering, and Intelligent Agents*, Vol. 1983, K. S. Leung, L.-W. Chan, and H. Meng, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 17–22, 2000.

[24] F. Dalpiaz, I. van der Schalk, S. Brinkkemper, F. B. Aydemir, and G. Lucassen, “Detecting terminological ambiguity in user stories: Tool and experimentation”, *Inf. Softw. Technol.*, Vol. 110, pp. 3–16, 2019, doi: 10.1016/j.infsof.2018.12.007.

[25] “Cortical.io - Next generation of AI business applications”. https://www.cortical.io/ (accessed Aug. 03, 2020).

[26] “Semantics of Business Vocabulary and Rules”. https://www.omg.org/spec/SBVR/About-SBVR/ (accessed Aug. 03, 2020).

[27] “Model Driven Architecture (MDA) | Object Management Group”. https://www.omg.org/mda/ (accessed May 02, 2020).

[28] C. D. Manning, M. Surdeanu, J. Bauer, J. R. Finkel, S. Bethard, and D. McClosky, “The Stanford CoreNLP natural language processing toolkit”, In: *Proc. of 52nd annual meeting of the association for computational linguistics: system demonstrations*, pp. 55–60, 2014.

[29] R. Menaha and VE. Jayanthi, “A Survey on Acronym–Expansion Mining Approaches from Text and Web”, *Smart Intelligent Computing and Applications, Singapore*, pp. 121–133, 2019, doi: 10.1007/978-981-13-1921-1_12.

[30] J. Pustejovsky, J. Castano, B. Cochran, M. Kotecki, M. Morrell, and A. Rumshisky, “Extraction and disambiguation of acronym-meaning pairs in medline”, *Medinfo*, Vol. 10, No. 2001, pp. 371–375, 2001.

[31] A. S. Schwartz and M. A. Hearst, “A simple algorithm for identifying abbreviation

definitions in biomedical text", *in Biocomputing 2003*, World Scientific, pp. 451–462, 2002.

[32] K. Taghva and J. Gilbreth, "Recognizing acronyms and their definitions", *Int. J. Doc. Anal. Recognit.*, Vol. 1, No. 4, pp. 191–198, 1999.

[33] S. Yeates, "Automatic Extraction of Acronyms from Text.", *in New Zealand Computer Science Research Students' Conf.*, 1999, pp. 117–124.

[34] L. S. Larkey, P. Ogilvie, M. A. Price, and B. Tamilio, "Acrophile: an automated acronym extractor and server", In: *Proc. of the fifth ACM conference on Digital libraries*, pp. 205–214, 2002.

[35] Y. Park and R. J. Byrd, "Hybrid text mining for finding abbreviations and their definitions", 2001.

[36] J. Liu, C. Liu, and Y. Huang, "Multi-granularity sequence labeling model for acronym expansion identification", *Inf. Sci.*, Vol. 378, pp. 462–474, 2017, doi: 10.1016/j.ins.2016.06.045.

[37] D. Sánchez and D. Isern, "Automatic extraction of acronym definitions from the Web", *Appl. Intell.*, Vol. 34, No. 2, pp. 311–327, 2011, doi: 10.1007/s10489-009-0197-4.

[38] D. Nadeau and P. D. Turney, "A Supervised Learning Approach to Acronym Identification", *in Advances in Artificial Intelligence*, Berlin, Heidelberg, pp. 319–329, 2005 doi: 10.1007/11424918_34.

[39] E. Adar, "SaRAD: A simple and robust abbreviation dictionary", *Bioinformatics*, Vol. 20, No. 4, pp. 527–533, 2004.

[40] J. T. Chang, H. Schütze, and R. B. Altman, "Creating an online dictionary of abbreviations from MEDLINE", J. Am. Med. Inform. Assoc., Vol. 9, No. 6, pp. 612–620, 2002.

[41] J. Xu and Y. Huang, "Using SVM to extract acronyms from text", *Soft Comput*., Vol. 11, No. 4, pp. 369–373, 2007.

[42] Z. Wu and M. Palmer, "Verbs semantics and lexical selection", In: *Proc. of the 32nd annual meeting on Association for Computational Linguistics*, pp. 133–138, 1994.

[43] P. Resnik, "Using information content to evaluate semantic similarity in a taxonomy", ArXiv Prepr. Cmp-Lg9511007, 1995.

[44] J. J. Jiang and D. W. Conrath, "Semantic similarity based on corpus statistics and lexical taxonomy", ArXiv Prepr. Cmp-Lg9709008, 1997.

[45] D. Lin, "Extracting collocations from text corpora", *in First workshop on computational terminology*, pp. 57–63, 1998.

[46] C. Leacock and M. Chodorow, "Combining local context and WordNet sense similarity for word sense identification. WordNet, An Electronic Lexical Database", *MIT Press*, 1998.

[47] S. Banerjee and T. Pedersen, "An adapted Lesk algorithm for word sense disambiguation using WordNet", In: *Proc. of International Conf. on Intelligent Text Processing and Computational Linguistics*, pp. 136–145, 2002.

[48] G. Hirst and D. St-Onge, "Lexical chains as representations of context for the detection and correction of malapropisms", *WordNet Electron. Lex. Database*, Vol. 305, pp. 305–332, 1998.

[49] G. C. Witt, Writing effective business rules a practical method. 2012.

[50] I. Graham, Business rules management and service oriented architecture: a pattern language. Chichester, England; Hoboken, NJ: John Wiley, 2006.