УДК 004.891.2

Misbakhul Munir Irfan Subakti[1, 2]
[1] Institute of Automation and Information Technologies,
Tambov State Technical University, Russia, Tambov,
[2] Department of Informatics, Institut Teknologi Sepuluh Nopember (ITS),
Indonesia, Surabaya
(Tel. (+7920)4992813, e-mail: yifana@gmail.com)

INTEGRATING CONTEXTUAL INFORMATION INTO ONTOLOGIES FOR READING MATERIAL CLASSIFICATION

Abstract. Reading Material Classification (RMC) classifies an unclassified text into the readability graded reading material based on its text readability. Recent approaches for RMC have used Natural Language Processing (NLP) methods such as machine-learning-based methods (e.g., Support Vector Machine, Multinomial Naïve Bayes and Latent Semantic Indexing) to overcome disadvantages of using the syntactic features, i.e., insufficiency for modeling the levels of text reading difficulty. Ontologies have been used for sharing and reusing knowledge, and perhaps supporting the inference. It will be used for RMC. Concepts and contexts are treated separately in ontologies. By only using basic NLP techniques such as stemming and word sense disambiguation, integrating contextual information into ontologies, i.e., Contextual Ontology (CO), is proposed which aimed to improve the ontology's performance and possibly other types of quality for RMC. From the evaluation experiments, we do not claim that our proposed method is better than machine-learning based RMC, our system performance is just on a par with them. Rather than beating them, we aimed to use RMC to show that integrating contextual information into ontologies (RMC-CO) provides a considerable benefit for ontologies than not integrate it (RMC-O). 1.56% and 2.11% improvements can be obtained for validation and testing data, respectively.

Keywords: contextual information, contextual ontology, machine learning, ontology, reading material classification.

1. Introduction

The process where an unclassified text is classified based on the difficulty level of its text readability for assigning appropriate reading material is called Reading Material Classification (RMC). A reliable and an automatic RMC is urgently needed in education and other domains. Even though in principle RMC is not confined to K-12 Education, for reasons of availability RMC for K-12 (primary and secondary) Education will be chosen for the test-bed. RMC has been actively researched by researchers, where syntactic features, e.g., word frequency, syllable and character counts per word and sentence length, have been used as the indicators of text readability. However, the levels of text reading difficulty cannot be modelled sufficiently. To overcome this drawback, machine-learning-based

techniques, e.g., Support Vector Machine (SVM), Multinomial Naïve Bayes and Latent Semantic Indexing (LSI), which used Natural Language Processing (NLP) methods in their methods, have been used in recent research.

Ontologies constituted by concepts as the main components along with relations, instances and axioms (Noy and Tu, 2003). Concepts themselves have been defined as the representation of a set of classes of entities or 'things' within a domain (Gruber, 1993). On the other hand, a context is defined as a situation within which something exists or happens (Peters, 2013). Past ontologies researchers have included context-based aspects which are mostly served as the additional layers above ontologies, e.g., they are less integrated with other components. For instance, Leppanen (2007) proposed a top-level ontology, namely the context-based enterprise ontology, where a context can be found in the sentence, conversation and action; and is categorised into the domains of purpose, actor, action, object, facility, location and time. The separation between concepts and contextual components motivated us for an idea in integrating contextual components and concepts for improving an ontology's performance and possibly other types of quality.

In our ontologies, a contextual component namely «contextual concept-variant» (which can be abbreviated to «concept-variant» for its brevity) is defined as a specific level of description about a concept. Concept-variant can be distinguished with «context» which has its ordinary meaning. A «concept variant» in our system application, may not be a variant of any of the particular «concepts» arising in that application. Any concept-variant is not necessarily a variant of one of the concepts, so concept-variants should be called potential or candidate concept-variants. However, we will omit the potential or candidate for brevity. Since a concept can be varied somewhat between different contexts, we will utilise this circumstance, i.e., having concept-variants integrated into ontologies (namely Contextual Ontology (CO)) to improve an ontology's performance and possibly other types of quality which will be applied in RMC as the test-bed. In processing the CO, some basic NLP techniques, e.g. stemming and word sense disambiguation will be used.

Given the source-text, i.e., the classified text served as the training data, concepts and concept-variants are extracted by concept extraction and concept-variant extraction, respectively, to build CO by CO extraction. Likewise, from the input-text (i.e., an unclassified text), its concepts and concept-variants, as well as its CO, also can be extracted. Two extracted information from the input-text and the source-text will be compared for RMC by matching procedures (i.e., concept matching, concept-variant matching and concept-concept-variant matching, respectively). If concepts and concept-variants are treated separately, then, it is ontology-based RMC,

i.e., RMC-O. If they are treated as an integrated one, then, it is CO-based RMC, i.e., RMC-CO.

K12Reader (K12Reader, 2017) provides reading material resources for K-12 Education from experienced reading teachers and curriculum specialists in the USA. Aside K12Reader dataset, a set of English essays written by high school students from the University of the Philippines Integrated School High School Division and Philippine Science High School (UPIS, 2017), is also used as the dataset for our experiments. The performance of our method will be compared to the Readability formulas (Readability Formulas, 2017) – a well-known statistics-based widely used to evaluate RMC, for the baseline of comparison. Then, RMC-O and RMC-CO will be compared to see the improvement.

The remainder of the paper is organised as follows. Section 2 provides a detailed description of how our system works. Section 3 explains the experiments for evaluating our system's performance before we conclude the paper in Section 4, along with our future works.

2. Proposed Method Description

2.1. *Automatic Extraction*: *Concept, Concept-Variant and Contextual Ontology Extractions*

A concept is defined as lexical shared and collocated words as well as their hyponyms and synonyms which are found in the source (i.e., a text or a number of source texts) of a given domain. This definition can be broadened into a structure, a tree and more complicated forms of these combinations (i.e., the combinations of word, structure and tree). The notion of «concept» adopted from Ahmad and Gillam (2005), where they defined «concepts» as a thesaurus of terms, each of which is expected to denote a concept. The words extracted by their word extraction are used as our concepts.

Bauer and Leake (2001, 2002) via their work, namely WordSieve, and Jain et al. (2013) which used RAKE (Rapid Automatic Keyword Extraction, proposed by Rose et al. (2010)), called the extracted words from their word extraction procedures as «contexts». We have considered a naming «contexts» produced as the outputs of their works is slightly inappropriate. We propose «contextual concept-variants» as a better name for these outputs. Contextual concept-variant (which can be abbreviated to «concept-variant» for convenience purposes) is defined as the frequently occurring word/term and a sequence of one or more words which provide a compact representation of the content of a text (or a number of source text) along with its weight found in the source, i.e., a text or a number of texts. This definition can be broadened into a structure, a tree and more complicated forms of these combinations (i.e., the combinations of word, structure and tree). A weight is defined as a measure of the evidence accumulated so far for a particular grade level.

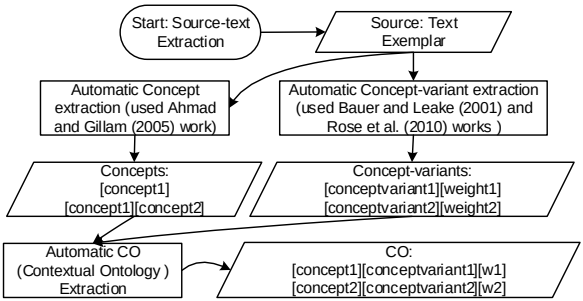

Figure 1. Source-text extraction

To obtain information for RMC, our system started by extracting the source-info from the source-text (figure 1), where concept and concept-variant extractions are performed. Likewise, the input-info extracted from the input-text (i.e., an unclassified text) so that it also contains concepts and concept-variants. Concept extraction used a work by Ahmad and Gillam (2005), and concept-variant extraction used a work by Bauer and Leake (2001), namely WordSieve, and a work by Rose et al. (2010), namely RAKE. RMC can be done by comparing the input-info and the source-info by matching procedures, i.e., concept, concept-variant and concept-concept-variant matching (section 2.2). For RMC-CO, concept-variants will be integrated into concepts to build CO, for both the input-info and the source-info. The weights produced by matching procedures can be used individually (i.e., for each concept, concept-variant and concept-concept-variant matching, respectively) or averaged/combined altogether. The final weight, i.e., a particular grade level with the highest weight will be chosen as the grade level assigned to the input-text.

Figure 2 shows a flow chart for the concept extraction. From the text, the frequency of each word's occurrences (f) is calculated as well as the weirdness index (w). A measure of the use of a word in special language compared to its use in a representative corpus of general language texts is represented by weirdness index (Equation 1).

$$\text{weirdness} = \frac{N_B f_T}{(1+ f_B) N_T} . \quad (1)$$

Where from a given the word, f_T is its frequency in our text, f_B is its frequency in British National Corpus (BNC). NT is the number of all words in our text. NB is the number of all words in BNC. From the period of 1960 – 1993, BNC contains more than 100 million words. To get the lexical shared words among the words in the folder of source-text, z-score for both f and w is calculated (equation 2) as a measure of disproportionately used words.

$$z_i(\text{word}) = \frac{(\text{word}_i - \overline{\text{word}})}{\sigma_{\text{word}}}. \quad (2)$$

Where z_i is the *i*-th z-score for word word_i, $\overline{\text{word}}$ and σ_{word} are the mean and standard deviation of word word, respectively (Ahmad and Gillam, 2005). The words whose z-score above a given threshold are extracted from the text as the concepts.

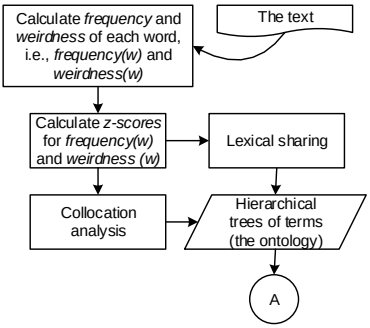

Figure 2. Concept extraction

The collocated words from Smadja (1993) are used, where they have the importance's position within their neighbourhoods, are also extracted as the concepts. Smadja has argued that significant collocates of a word are within a neighbourhood of five words either side of the word (five words to the left, L1-L5, and five words to the right, R1-R5). Metrices for quantifying the strength of the collocation are the one that isolates peakedness (U-score) of the collocation in the various positions of the neighbourhood, along with the z-score. Peakedness is described by the equation 3 as in the following.

$$\text{U-score} = U_i = \frac{\sum_{j=1}^{10} (p_i^j - \overline{p}_i)^2}{10}. \quad (3)$$

Where p_i^j is a frequency of word word_i with word word such that they *j* words apart, $\overline{p}_i$ is the mean of word word (Smadja, 1993). The words whose U-score above a given threshold are extracted from the text as the concepts. The words obtained from lexical sharing and collocation analysis procedures are the concepts. Figure 2 – 4 show two connectors: A and B. Connector A means the concepts extracted in figure 2 will goes to CO extraction in figure 4, along with connector B, i.e., the extracted concept-variants (figure 3).

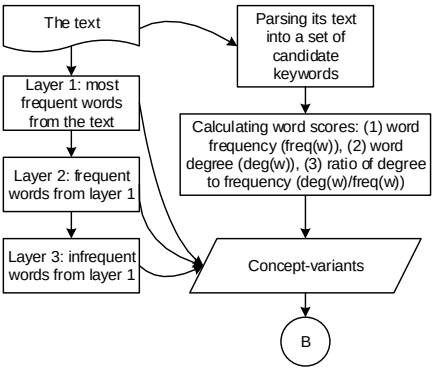

Figure 3. Concept-variant extraction

Figure 3 shows a flow chart for concept-variant extraction. WordSieve network consists of three small, interdependent levels of nodes. Level 1 is sensitised to the words that are currently in the text; level 2 keeps a record of the words that have tended to occur at different times and only slowly forget them, and level 3 works in the opposite direction of level 2. Level 3 keeps a low level of activation value until the word stops occurring and increase this value for as long as the word does not occur. These three levels then produce our concept-variants.

RAKE is used to get more concept-variants, where a set candidate keywords are obtained by parsing the source-text. The text from the source-text is split into an array of words by the specified word delimiters, then split into sequences of contiguous words at phrase delimiters and stop word positions. Words within a sequence are assigned the same position in the text, and together, they are treated as a candidate keyword. After every candidate word is found and the graph of word co-occurrences (i.e., the rows and columns are filled with candidate keywords, so that co-occurrence of keywords can be evaluated) is complete, a score is calculated for each candidate keyword, and it is defined as the sum of its member word scores. Equation 4 shows metrices for calculating the word scores, based on the degree and frequency of word vertices in the graph.

$$\begin{gathered} \text{freq(word)} = \text{word frequency} \\ \text{deg(word)} = \text{word degree} \\ \text{deg(word)} / \text{freq(word)} = \text{ratio of degree of frequency} \end{gathered} \tag{4}$$

Where freq(word) is the occurrence number of word word in the text and deg(word) is the sum of the co-occurrence numbers of a word word in its row in the word co-occurrence graph. Deg (word) favours words that often occur and in longer candidate words. Words that frequently occur regardless of the number of words with which they co-occur are favoured by freq (word). Words that predominantly occur in longer candidate words are favoured by deg (word) / freq (word). The score for each candidate word is computed as the sum of its member word scores.

Figure 4 shows CO extraction, where concept-variants (i.e., extracted by concept-variant extraction) will be integrated with concepts (i.e., extracted by concept extraction). Additional information, i.e., organisational features such as the folder structure of a number of text files in the source-text, is used to build CO. WordNet (WordNet, 2017) is used for the electronic thesaurus, to find the hyponyms and synonyms of a word. Sangers et al. (2013) also used WordNet for finding monosemous words to establish a starting context. Basic techniques of NLP such as stemming and word sense disambiguation is used in CO extraction. Algorithm 1 shows our proposed CO extraction algorithm.

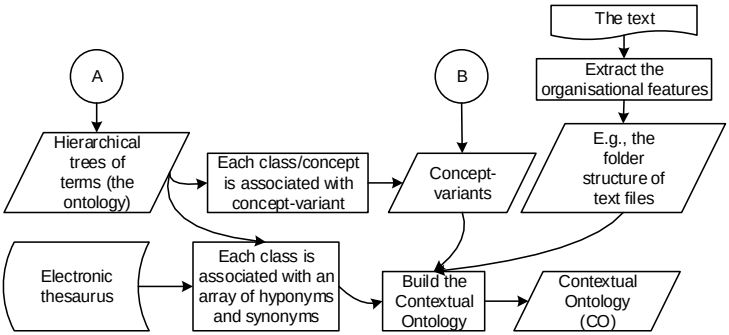

Figure 4. CO extraction

Algorithm 1. Contextual Ontology (CO) Extraction

Input: concepts, concept-variants and organisational features (e.g., the folder structure of a number of text files in the source-text).

Obtaining concepts and concept-variants

1. Get concepts from concept extraction and save them to ConceptList.

E.g., ConceptList : = {[conceptA], [conceptB], …, [conceptN]}

2. Find a given number of hyponyms and synonyms of each concept from ConceptList, with the support of an electronic thesaurus, e.g., WordNet.

FOR each concept in ConceptList

Get a given number of hyponyms and synonyms of a concept, and save them to ConceptList if any.

E.g., ConceptList : = {[conceptA] ([hyponymA1], [hyponymA2], …, [hyponymAN], [synonymA1], [synonymA2], …, [synonymAN]), [conceptB] ([hyponymB1], [hyponymB2], …, [hyponymBN], [synonymB1], [synonymB2], …, [synonymBN]), …, [conceptM] ([hyponymM1], … , [synonymMN])}

3. Get concept-variants from concept-variant extraction and save them to ConceptVariantList.

E.g., ConceptVariantList : = {[conceptvariantA][weightA], [conceptvariantB][weightB], …, [conceptvariantN][weightN]}

CO before updating

4. Build the contextualised concept (CO) from ConceptList. A node in CO, namely CONode, is defined as [concept][conceptvariant][weight].

FOR each concept in ConceptList
Get a concept and add them to CONodes.
/* CONode = concept */
CONode : = [conceptA][][]
/* Add a new CONode to CONodes */
CONodes : = CONodes ∪ CONode

5. Build the CO from ConceptVariantList. Based on CONodes, check whether a concept-variant already exists in CONodes.

FOR each concept-variant in ConceptVariantList
IF a concept-variant is not in CONodes, assign a concept-variant to be a new CONode
/* CONode = concept-variant */
CONode : = [][conceptvariantA][weightA]
/* Add a new CONode to CONodes */
CONodes : = CONodes ∪ CONode

Updating CO

6. Get the organisational features such as the folder/directory structure of a number of text files in the source-text, and save it to FolderNameList. A folder name is important because it represents the class/group/categorisation of the texts, e.g., the grade levels.

FOR each concept in ConceptList and each concept-variant in ConceptVariantList
Save each concept to the folder name it belongs.
Count a number of folder/directory name from each concept and concept-variant, and save these numbers along with their folder names.
Save all the information above to FolderNameList.

FOR each folder name in FolderNameList
Get the percentage of each number of a folder name against the total number of all folder names.
/* 1st case of updating */

7. For each concept in ConceptList, find any concept-variant (from ConceptVariantList) whose the same word, if it exists, and populate a new CONode.

FOR each concept in ConceptList
IF a concept shared the same word with a concept-variant in ConceptVariantList, join a concept-variant to a concept together as a new CONode.

/* Join a concept and a concept-variant */
[conceptA][conceptvariantA][weightA] : = [conceptA] +
[conceptvariantA][weightA]
/* CONode = concept + conceptvariant */
CONode : = [conceptA][conceptvariantA][weightA]
/* Add a new CONode to CONodes */
CONodes : = CONodes ∪ CONode

/* 2nd case of updating */

8. If from the step 7 above, a CONode has an empty concept-variant and an empty weight (e.g., [conceptA][][]), fill the empty concept-variant with the folder name of its concept (from FolderNameList) and also fill the empty weight with the percentage of the folder name.

FOR each CONode in CONodes
IF concept-variant and the weight of a CONode are empty, e.g., [conceptA][][]
Fill the empty concept-variant with the folder name of its concept (from FolderNameList).
Fill the empty weight with the percentage of the folder name.
E.g., [conceptA][][]. [conceptA][classA][weightA], [conceptA][classB] [weightB], [conceptA][classC][weightC], …, [conceptA][classN][weightN]

/* 3rd case of updating */

9. Based on ConceptList, by using Semantic Relatedness Metrics (SRM) (WS4J, 2017) to calculate the similarity between a concept and the folder name's concepts and concept-variants. The information about the folder names are obtained from FolderNameList, and only a limited amount of concepts and concept-variants for each folder name will be used. Populate a new CONode by combining concepts and the folder names according to their SRM. As a result, then all of the CONodes now had grade levels (i.e., it is a folder name represented a grade level) as a concept-variant, and it is put preceded the weight, i.e., [concept][conceptvariant] [grade][weight].

FOR each concept in ConceptList
Get a concept, and calculate the SRM between this concept and a limited number of concepts as well as concept-variants of the folder names (i.e., they represent classes/grade levels) from FolderNameList.
/* Calculate SRM */
E.g., [weightA]. SRM between [conceptA] and concepts & concept-variants from [gradeA]
[weightB] SRM between [conceptA] and concepts & concept-variants from [gradeB]

/* Populate a new CONode */
/* by filling the empty weight with the percentage of the folder name */
E.g., [conceptA][][]
[conceptA][conceptvariantA][gradeA][weightA],
[conceptA][conceptvariantA][gradeB][weightB],
[conceptA][conceptvariantA][gradeC][weightC], …,
[conceptA][conceptvariantA][gradeN][weightN]

/* 4th case of updating */
10. As an additional feature, concept-variants from ConceptVariantList can be combined with a concept from the ConceptList.

FOR each concept-variant in ConceptVariantList
FOR each concept in ConceptVariantList
Combine a concept with concept-variants.
E.g., [conceptA][conceptvariantA][weightA] : = [conceptA] ∪ [conceptvariantA][weightA]
[conceptA][conceptvariantB][weightB] : = [conceptA] ∪ [conceptvariantB][weightB]
[conceptA][conceptvariantN][weightN] : = [conceptA] ∪ [conceptvariantN][weightN]

Output: CONodes, i.e., Contextual Ontology (CO).

2.2. *Matching Procedures*: *Concept, Concept-Variant and Concept-Concept-Variant Matchings*

Concept matching. RMC-O: the input-info will be matched with the source-info's concepts. RMC-CO: the input-info will be matched with the source-info's [concepts]. The partial matched is allowed. A weight obtained is simply 0 (no matched) or 1 (it is matched).

Concept-variant matching. RMC-O: the input-info will be matched with the source-info's concept-variants. RMC-CO: the input-info will be matched with the source-info's [concept-variants]. The partial matched is allowed. A weight obtained is simply 0 (no matched) or 1 (it is matched).

Concept-concept-variant matching. RMC-O: the input-info will be matched with the source-info's concepts and concept-variants. RMC-CO: the input-info will be matched with the source-info's [concepts] and [concept-variants]. The partial matched is allowed. A weight obtained is the gradation of 0 (no matched at all) to 1 (it is matched completely).

An example of matching procedures for RMC-CO as follows: when the source-info has {…, [grand][canyon][G7][0.62], [grand][villa][estate][G8][0.77], [grand][canyon][state][G9-10][0.66], …} and the input-

info has {…, [canyon][Arizona][0.20], …} then by concept matching, there are two cases are matched: a particular concept [canyon] is matched with both [grand][canyon][G7][0.62] and with [grand][canyon][state] [G9-10][0.66]. With [grand][canyon][G7][0.62], the weight assigned is (0.62 + 0.20) / 2 = 0.41 for concept-variant [G7], and with [grand][canyon] [state][G9-10][0.66], the weight assigned is (0.66 + 0.20) / 2 = 0.43 for concept-variant [G9-10]. Since 0.43 > 0.41, then from [grand][canyon] [state][G9-10][0.66] the particular result [G9-10] (i.e., Grade 9-10), will be assigned to the input-info if we want to use only concept matching for RMC. By concept-variant matching, concept-variant [Arizona] does not match with any particular CONode in CO, so that the weight assigned are 0 (zero) for all of [G7], [G8] and [G9-10]. For concept-concept-variant matching, the result is similar to of concept matching, since only concept [canyon] can be matched and there is no match for concept-variant [Arizona], i.e., for concept-variant [G7], [G8] and [G9-10], the weights 0.41, 0 and 0.43 are assigned to the input text, respectively.

3. The Experiments

K12Reader dataset has 186 files in the 8-grade levels. There are 80 files used in cross-validation, where those 80 files are not yet differentiated into training and validation sets. In each iteration of cross-validation, a random partitioning into 10 subsets has done, taking one of the ten to be the validation subset. The 10 results from the folds then can be averaged for producing a single estimation. UPIS dataset has 399 files in the 6-grade levels. There are 240 files used in cross-validation. In each iteration of cross-validation, a random partitioning into 10 subsets has done, taking one of the ten to be the validation subset. The 10 results from the folds for K12Reader and UPIS, respectively, can be averaged for producing a single estimation. The rest of the files from the test set of both datasets (i.e., 106 and 159 files for K12Reader and UPIS, respectively) will be used for our system's evaluation.

3.1. *RMC-O*

An ontology (i.e., concepts and concept-variants are treated separately, they are not integrated) is evaluated. Based on the validation data, figure 5–6 shows all the results from the 10-fold cross-validation for K12Reader and UPIS, respectively, which produced by Readability Formulas (RF) (Readability Formulas, 2017) and RMC-O, respectively. The performance accuracy averages of RMC-O are better than of RF, i.e., 18.75% against 10.89% (K12Reader) and 63.75% against 19.94% (UPIS). Likewise, for the testing data, the results can be seen in figure 7–8, where the performance accuracy averages of RMC-O are better than of RF, i.e. 19.15% against 4.99% (K12Reader) and 68.43% against 20.31% (UPIS). The performance

No	Experiment	Accuracy	
		Readability Formulas	RMC-O
1	1st fold	21.43%	12.50%
2	2nd fold	14.29%	25.00%
3	3rd fold	16.07%	12.50%
4	4th fold	8.93%	0.00%
5	5th fold	5.36%	12.50%
6	6th fold	5.36%	12.50%
7	7th fold	12.50%	25.00%
8	8th fold	12.50%	25.00%
9	9th fold	7.14%	25.00%
10	10th fold	5.36%	37.50%
Average		**10.89%**	**18.75%**

Figure 5. RMC-O: K12Reader, validation data

No	Experiment	Accuracy	
		Readability Formulas	RMC-O
1	1st fold	18.45%	66.67%
2	2nd fold	25.00%	62.50%
3	3rd fold	22.62%	70.83%
4	4th fold	19.05%	62.50%
5	5th fold	19.05%	62.50%
6	6th fold	17.26%	75.00%
7	7th fold	26.19%	58.33%
8	8th fold	21.43%	62.50%
9	9th fold	13.10%	50.00%
10	10th fold	17.26%	66.67%
Average		**19.94%**	**63.75%**

Figure 6. RMC-O: UPIS, validation data

No	Experiment	Accuracy	
		Readability Formulas	RMC-O
1	1st fold	4.99%	14.15%
2	2nd fold		28.30%
3	3rd fold		14.15%
4	4th fold		20.75%
5	5th fold		20.75%
6	6th fold		9.43%
7	7th fold		16.98%
8	8th fold		23.58%
9	9th fold		19.81%
10	10th fold		23.58%
Average		**4.99%**	**19.15%**

Figure 7. RMC-O: K12Reader, testing data

No	Experiment	Accuracy	
		Readability Formulas	RMC-O
1	1st fold	20.31%	67.92%
2	2nd fold		70.44%
3	3rd fold		72.96%
4	4th fold		67.92%
5	5th fold		69.18%
6	6th fold		71.07%
7	7th fold		64.15%
8	8th fold		69.81%
9	9th fold		69.18%
10	10th fold		61.64%
Average		**20.31%**	**68.43%**

Figure 8. RMC-O: UPIS, testing data

accuracy of RMC-O's result for UPIS is far better (it is more than triple, 68.43% against 19.15%) than that of K12Reader. We suggest that because of the more data are provided for UPIS than for K12Reader, and also more cohesive topics and words are available for UPIS than for K12Reader, then it will result better. However, more investigations are needed to understand this phenomenon.

3.2. *RMC-CO*

CO (i.e., an integration of concepts and concept-variants, so that concepts and concept-variants are treated as an integral part) is evaluated. Based on the validation data, figure 9–10 shows all folds' results for validation data of K12Reader and UPIS, respectively, produced by RF and RMC-CO, respectively. The performance accuracy averages of RMC-CO

are better than of RF, i.e., 20.0% against 10.89% (K12Reader) and 65.42% against 19.94% (UPIS). Likewise, for the testing data, the results can be seen in figure 11–12, where the performance accuracy averages of RMC-CO are better than of RF, i.e., 21.6% against 4.99% (K12Reader) and 70.31% against 20.31% (UPIS). The performance accuracy of RMC-CO's result for UPIS if far better (it is more than triple, 70.31% against 21.60% for the testing data, and it is also more than triple, 65.42% against 20.0% for validation data) than that of K12Reader. We suggest that is because the more data of training and validation are provided for UPIS than for K12Reader, and also more cohesive topics and words are available for UPIS than for K12Rader, then the result will be better than otherwise. However, once again, more investigations are needed to understand this phenomenon.

No	Experiment	Accuracy	
		Readability Formulas	RMC-CO
1	1st fold	21.43%	25.00%
2	2nd fold	14.29%	25.00%
3	3rd fold	16.07%	12.50%
4	4th fold	8.93%	0.00%
5	5th fold	5.36%	12.50%
6	6th fold	5.36%	12.50%
7	7th fold	12.50%	25.00%
8	8th fold	12.50%	25.00%
9	9th fold	7.14%	25.00%
10	10th fold	5.36%	37.50%
Average		**10.89%**	**20.00%**

Figure 9. RMC-CO: K12Reader, validation data

No	Experiment	Accuracy	
		Readability Formulas	RMC-CO
1	1st fold	18.45%	58.33%
2	2nd fold	25.00%	70.83%
3	3rd fold	22.62%	66.67%
4	4th fold	19.05%	66.67%
5	5th fold	19.05%	62.50%
6	6th fold	17.26%	79.17%
7	7th fold	26.19%	62.50%
8	8th fold	21.43%	70.83%
9	9th fold	13.10%	50.00%
10	10th fold	17.26%	66.67%
Average		**19.94%**	**65.42%**

Figure 10. RMC-CO: UPIS, validation data

No	Experiment	Accuracy	
		Readability Formulas	RMC-CO
1	1st fold	4.99%	16.98%
2	2nd fold		32.08%
3	3rd fold		18.87%
4	4th fold		20.75%
5	5th fold		24.53%
6	6th fold		12.26%
7	7th fold		18.87%
8	8th fold		23.58%
9	9th fold		21.70%
10	10th fold		26.42%
Average		**4.99%**	**21.60%**

Figure 11. RMC-CO: K12Reader, testing data

No	Experiment	Accuracy	
		Readability Formulas	RMC-CO
1	1st fold	20.31%	69.81%
2	2nd fold		76.10%
3	3rd fold		72.96%
4	4th fold		72.33%
5	5th fold		69.18%
6	6th fold		72.96%
7	7th fold		65.41%
8	8th fold		74.21%
9	9th fold		69.81%
10	10th fold		60.38%
Average		**20.31%**	**70.31%**

Figure 12. RMC-CO: UPIS, testing data

3.3. *Contextual Information Significance*

The number of files is taken into account to make a fair comparison, i.e., the result will be in weighted averages. Figure 13–14 shows a comparison of the results to show how significance the integration of contextual information into ontologies to improve their performance accuracies for validation and testing data, respectively. Our proposed method beats the performance accuracy of RF in both validation and testing data. For the validation data, without integrating contextual information, RMC-O obtained 52.5%, a gap of 34.82%, compared to RF, i.e., 17.68%. By integrating contextual information, RMC-CO able to get 54.06%, a gap of 36.38%, compared to RF, i.e., 14.18%. It means there is an increment of 1.56% (i.e., 54.06%-52.5%). When it comes to the testing data, the performance accuracy improvement can also be seen even better as follows: without integrating contextual information, RMC-O obtained 48.72%. By integrating contextual information, RMC-CO able to get 50.83%, an increment about 2.11%, compared to RMC-O.

3.4. *CO-based RMC and Machine-learning-based RMC Comparison*

By using UPIS dataset, RMC-CO obtained 65.42% and 70.31% performance accuracies for validation and testing data, respectively. As for the comparison purposes, the best results from machine-learning-based RMC as follows: 63%-79% (Multinomial Naïve Bayes, Collins-Thompson and Callan (2004)), 75.4% (Expectation Minimisation, Si and Callan (2001)), 86.81% (validation data) and 87.16% (testing data) by SVM (Liu et al, 2004), 75% precision and 87% recall (n-gram language models of SVM, Schwarm and Ostendorf (2005)), 79.72% (Decision Tree, Wang (2006)), 76.18% (Naïve Bayes, Wang (2006)), 84.06% (SVM, Wang (2006)) and 88% (LSI, Landauer (2011)). From these results comparisons, we do not claim that our proposed method is better than machine-learning-based RMC, our system performance is just on a par with them.

No	Dataset	Number of Files	Accuracy: Validation Data		
			Readability Formulas	RMC-O	RMC-CO
1	K12Reader	80	10.89%	18.75%	20.00%
2	UPIS	240	19.94%	63.75%	65.42%
Weighted Average			**17.68%**	**52.50%**	**54.06%**

Figure 13. Weighted average comparison: validation data

No	Dataset	Number of Files	Accuracy: Testing Data		
			Readability Formulas	RMC-O	RMC-CO
1	K12Reader	106	4.99%	19.15%	21.60%
2	UPIS	159	20.31%	68.43%	70.31%
Weighted Average			**14.18%**	**48.72%**	**50.83%**

Figure 14. Weighted average comparison: testing data

4. Conclusion and Future Work

RMC-CO integrated contextual information into ontologies, i.e., CO, while RMC-O does not have such integrated ontology. By using weighted average calculation, 52.5% and 54.06% performance accuracies can be produced by RMC-O and RMC-CO, respectively, for the validation data. For the testing data, 48.72% and 50.83% can be produced by RMC-O and RMC-CO, respectively. About 1.56% (i.e., 54.06%-52.5%) and 2.11% (50.83%-48.72%) performance accuracy improvement can be obtained for validation and testing data, respectively; it shows that CO does improve the performance of ontologies.

Even 1.56% improvement for validation data is a considerable benefit for our application since the accuracies produced by RF are very poor, i.e., 17.68% and 14.18% for validation and testing data, respectively. So, even without the contextual information, RMC-O able to beat RF by 34.82% (i.e., 52.5%-17.68%) and 34.53% (i.e., 48.72%-14.18%) for validation and testing data, respectively. By integrating CO into ontologies, RMC-CO able to increase the gaps by 36.38% and 36.65% for validation and testing data, respectively.

Giving the description above, it can be concluded that thoroughly integrating contextual information into ontologies, rather than having it separated, is beneficial for ontologies.

More investigation about relationship within concepts and contextual information in CO can be performed to find any aspect which can be beneficial for reorganising CO itself and also for finding any potential advantages aside sharing, reusing and classifying tasks which just successfully demonstrated above.

References

1. **Ahmad, K.** and Gillam, L. (2005) Automatic Ontology Extraction from Unstructured Texts. Lect. Notes in Comp. Sci., 3761. – P. 1330 – 1346.

2. **Bauer, T.** and Leake, D. B. (2001) WordSieve: A Method for Real-Time Context Extraction. Lect. Notes in Comp. Sci., 2116. – P. 30 – 44.

3. **Bauer, T.** and Leake, D. B. (2002) Using Document Access Sequences to Recommend Customized Information. IEEE Intelligent Systems 17 (6). IEEE Educational Activities Department Piscataway: NJ, USA. – P. 27 – 33.

4. **Collins-Thompson**, K. and Callan, J. (2004) A Language Modelling Approach to Predicting Reading Difficulty. In: Proceedings of HLT/NAACL 2004. – P. 193 – 200.

5. **Gruber, T. R.** (1993) A Translation Approach to Portable Ontologies. Knowledge Acquisition, 5 (2). – P. 199 – 220.

6. **Jain, P.**, Bergen, A., Castaneda, L. and Muller, H. A. (2013) PALTask Chat: A Personalized Automated Context-Aware Web-Resources Listing Tool. In: Proceedings of 2013 IEEE Ninth World Congress on Services. Santa Clara, CA: IEEE. – P. 154 – 157.

7. **K12Reader**. (2017) Available: http://www.k12reader.com [Accessed 14.09.2017].

8. **Landauer, T. K.** (2011) Pearson's Text Complexity Measure. Pearson's White Papers. – 2011.

9. **Leppänen, M.** (2007) A Context-Based Enterprise Ontology. In: Abramowicz W. (ed.) 10th International Conference on Business Information Systems (BIS 2007), Poznan, Poland, 25 – 27.4.2007, Lect. Notes in Comp. Sci., 4439. Berlin, Heidelberg: Springer-Verlag. – P. 273 – 286.

10. **Liu, X.-Y.,** Croft, W. B., Oh, P. and Hart, D. (2004) Automatic Recognition of Reading Levels from User Queries. In: Proceedings of Sheffield SIGIR 2004. The 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Sheffield, UK. – P. 548–549.

11. **Noy, N. F.** and Tu, S. W. (2003) Developing Medical Informatics Ontologies with Protégé. In: Proceedings of the American Medical Informatics Association Annual Symposium (AMIA'03).

12. **Peters, P.** (2013) The Cambridge Dictionary of English Grammar. Cambridge University Press, ISBN-10: 0521863198, ISBN-13: 978-0521863193.

13. **Readability** Formulas. (2017) Available: http://www.readability-formulas.com [Accessed 14.09.2017].

14. **Rose, S.,** Engel, D., Cramer, N. and Cowley, W. (2010) Automatic Keyword Extraction from Individual Documents. Text Mining: Applications and Theory. John Wiley & Sons. – P. 1 – 20.

15. **Sangers, J.,** Frasincar, F., Hogenboom, F. and Chepegin, V. (2013) Semantic Web Service Discovery Using Natural Language Processing Techniques. Expert Systems with Applications, 40 (11). Tarrytown, NY, USA: Pergamon Press, Inc. – P. 4660 – 4671.

16. **Schwarm, S. E.** and Ostendorf, M. (2005) Reading Level Assessment Using Support Vector Machines and Statistical Language Models. In: Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics, Assoc. for Comp. Linguistics. – P. 523 – 530.

17. **Si, L.** and Callan, J. (2001) A Statistical Model for Scientific Readability. In: Proceedings of the 2001 ACM CIKM. 10th International Conference on Information and Knowledge Management, Atlanta, GA, USA. – P. 574 – 576.

18. **Smadja, F.** (1993) Retrieving Collocations from Text: Xtract. Computational Linguistics, 19(1). Oxford University Press. – P. 143 – 178.

19. **University** of the Philippines Integrated School (UPIS). (2017) Available: http://www.upis.upd.edu.ph [Accessed 14.09.2017].

20. **Wang, Y.-L.** (2006) Automatic Recognition of Text Difficulty from Consumers Health Information. Computer-Based Medical Systems, 2006. CBMS 2006. 19th IEEE Int'lSymposium on, 2006. – P. 131 – 136.

21. **WordNet** Similarity for Java (WS4J). (2017) Available: https:// code.google.com/p/ws4j [Accessed 14.09.2017].

22. **WordNet.** (2017) Available: https://wordnet.princeton.edu/ [Accessed 14.09.2017].