Computer Science • 22(2) 2021 https://doi.org/10.7494/csci.2021.22.2.3868

Reza Fauzan
Daniel Siahaan
Siti Rochimah
Evi Triandini

NOVEL APPROACH TO AUTOMATED BEHAVIORAL DIAGRAM ASSESSMENT USING LABEL SIMILARITY AND SUBGRAPH EDIT DISTANCE

Abstract *The Unified Modeling Language (UML) is one of the standard languages that are used in modeling software; therefore, UML is widely taught in many universities. Generally, teachers assign students to build UML diagram designs based on a predetermined project; however, the assessment of such assignments can be challenging, and teachers may be inconsistent in assessing their students' answers. Thus, automated UML diagram assessment becomes essential to maintaining assessment consistency. This study uses a behavioral diagram as the object of research, since it is a commonly taught UML diagram. The behavioral diagram can show a dynamic view of the software. This study proposes a new approach to automatically assessing the similarity of behavior diagrams as reliably as experts do. We divide the assessment into two portions: semantic assessment, and structural assessment. Label similarity is used to calculate semantic assessment, while subgraph edit distance is used to calculate structural assessment. The results suggest that the proposed approach is as reliable as an expert in assessing the similarity between two behavior diagrams. The observed agreement value suggests a strong agreement between the use of experts and the proposed approach.*

Keywords automated assessment, behavioral diagram, label similarity, similarity assessment, subgraph edit distance, Unified Modeling Language

Citation Computer Science 22(2) 2021: 191–207

1. Introduction

Unified Modeling Language (UML) diagrams are diagrams that are commonly used for document software development life cycles. Notably, measuring the similarity of UML diagrams can improve the quality of the software development life cycle [1]. Diagram similarity measurement can be grouped into three applications in the learning process: software component reuse, software design plagiarism checking, and software design assessment. A behavior diagram is a portion of a UML diagram that provides a dynamic view of a software system [30]. The behavior diagrams commonly used to measure similarity include state diagrams [2, 3] and sequence diagrams [4, 27, 29, 31]. Notably, the present study uses sequence diagrams as research objects in behavioral diagrams [23]. Sequence diagrams can show the flow and interaction between objects based on system requirements [8]. This study represents an improvement from previous studies [29,31] that discuss the label similarity in sequence diagrams semantically.

Several studies have investigated the similarity of behavioral diagrams. These studies were initiated by Park [22], Salami [28], and Adamu [4] by using sequences of messages between objects in a sequence diagram. The aforementioned authors used only message sequences from sequence diagrams to build graphs. In this case, a built graph is called a message object order graph (MOOG) and represents the structure of a sequence diagram. A built graph has vertices in the form of sequential objects that are based on the order of the messages. The three previous studies aimed at increasing the efficiency of the reuse process in the repository and showed that structural similarity is useful for calculating the similarity between two sequence diagrams. However, structural similarity is not sufficient for measuring similarity, since it must be combined with semantic similarity to measure similarity based on meaning. Siahaan [29] measured the similarity between two sequence diagrams using the label information from a diagram. The obtained label information was compared semantically using natural language processing. This research suggests that label similarity can also be a good way of measuring the similarity of two sequence diagrams.

Based on previous studies, we can conclude that experts observe diagrams based on their meanings and structures. Experts observe meanings based on the label similarity in a diagram, which is calculated as a semantic assessment. Moreover, they observe structure based on the relationships between elements in a diagram as a structural assessment.

In the present study, we focus on measuring the similarity to behavioral diagram assessment in learning. In the field of software engineering, teachers often cover behavioral diagrams in software design. Thus, teachers often ask students to design software based on a predetermined case; the teachers then assess their students' assignments based on an answer key. However, teachers often find it challenging to maintain consistency in assessing such assignments [5,17]. This issue is caused by two factors [6]. First, teachers may use subjective assessment to assess different students due to fatigue. Second, assessment criteria may differ between teachers. Therefore, an approach that can automatically assess the similarity of two behavioral diagrams

is needed for assessment consistency to be maintained. Previous studies only used behavioral diagrams and did not assess similarity so that the weights of an expert's perspective could differ from one another. Furthermore, previous studies also measured similarity separately for semantic assessment and structural assessment.

This study proposes an approach that combines semantic and structural assessment to assess the similarity between two behavioral diagrams. As previously explained, the behavioral diagram used in this study is the sequence diagram. Notably, the label similarity between two diagrams determines the semantic assessment, while structural assessment is determined by the subgraphs of graphs that are built based on the diagram. To maintain consistency in the assessment, the proposed method is as reliable as an expert in assessing the similarity of two diagrams. We also show the extent to which an expert considers semantic or structural aspects in assessing similarity. This contribution to the semantic assessment approach provides an appropriate weighting value based on expert methods of assessing similarity. We also use natural language processing [16,19] to measure label similarity semantically. The contribution of the present study to the structural similarity assessment approach is the creation of a new graph model known as the UML Common Graph. We use subgraph edit distance [9,14,25] to measure structural similarity.

2. Label Similarity

The main idea of label similarity is to compare two sequence diagrams based on the label information found in the diagrams. The label information in sequence diagrams consists of object information and message information.

2.1. Semantic Similarity

Semantic similarity is used to assess the similarity between two phrases in two sequence diagrams. For example, we can assess the similarity of one message from an object in the first sequence diagram to a message from the object in the second sequence diagram by assessing the similarity of the label information. Similarity assessment for two labels uses natural language processing [16,21]. Natural language processing evaluates the similarity based on the semantic meaning of the compared terms.

Our previous studies [10,11,31] used natural language processing as part of assessing label similarity semantically. Notably, the present study uses the same flow to assess label similarity semantically. The applied similarity assessment flow involves tokenization, part of speech (POS) tagging, stop word removal, lemmatization, and cosine similarity. One word in a sequence diagram can consist of several words. For example, a message named *enterOption* must be broken down into *enter* and *option* via tokenization. Thereafter, the words are tagged. If these words have stop words, the stop words will be removed. Then, the words are changed to the first form using lemmatization. Finally, the first and second sets of words are assessed using cosine

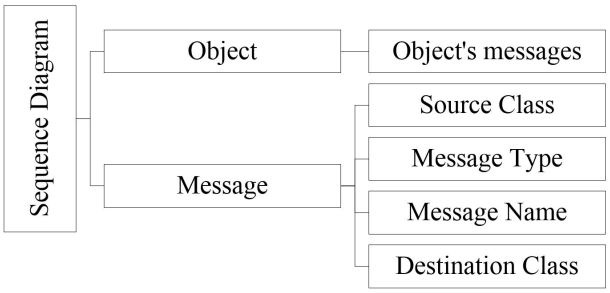

Figure 1. NLP in UML Diagram Semantic Similarity Assessment

similarity [7]. Detailed calculations between two words in cosine similarity are then performed using Wu Palmer [13] and WordNet [12]. The similarity of meaning between two words is calculated using Wu Palmer based on the distance between words in WordNet.

2.2. Semantic Assessment on Sequence Diagrams

Semantic assessment between two sequence diagrams is performed based on the elements that they contain. Label information from one component in the first diagram will be compared to label information from the same component in the second diagram (e.g., messages will be compared to messages). However, the message will not be compared with the name of the object or class. Based on a previous study [31], label information sequence diagrams are divided into two major groups: object, and message.

Figure 1 shows that sequence diagrams are assessed based on the label information of the object and the messages contained in each diagram. The process of taking semantic information from a sequence diagram has been explained in previous studies [31]. Each object from the first sequence diagram is compared to each object from the second sequence diagram based on the information messages that pass through the object. Then, each message from the first sequence diagram is compared to each message from the second sequence diagram. A message can have a source class, message type, message name, and destination class. The formula for assessing the label semantically between two sequence diagrams is presented in Equation 1.

$$semSim(d_1, d_2) = ((1 - \rho_{sem}) \times oSim(d_1, d_2)) + (\rho_{sem} \times msSim(d_1, d_2)) \quad (1)$$

The semantic assessment of first sequence diagram d_1 and second sequence diagram d_2 is $semSim$. ρ_{sem} is used to assign an importance to each semantic element in the assessment. The value of ρ ranges from 0 to 1. The elements assessed include object similarity ($oSim$) and message similarity ($msSim$). Equation 2 is the formula

for assessing the similarity of objects between two sequence diagrams. Equation 3 is a formula for assessing the similarity of messages between two sequence diagrams.

$$oSim(d_1, d_2) = \frac{2 \times (\sum_{k=1}^{\min(|MSN_1|,|MSN_2|)} changePivot(\sum_{i=1}^{|MSN_1|} \sum_{j=1}^{|MSN_2|} cosineSim(msn_i, msn_j)))}{|MSN_1| + |MSN_2|} \tag{2}$$

Equation 2 involves the implementation of the greedy algorithm to semantically assess the object's collection (MSN_1) in sequence diagram d_1 to the object's collection (MSN_2) in sequence diagram d_2. Each object *msn* contains a label collection of messages that pass through it. The greedy algorithm works by finding the optimal value of each pair of similarities. In the search for optimal values, *changePivot* eliminates optimal values. Algorithm 1 explains the flow of *changePivot*.

Algorithm 1: changePivot

Input: two-dimension matrix and pivot/coordinate maximum value (x,y)
Output: changed matrix
1 Select pivot
2 $M(x, :) = 0$
3 $M(:, y) = 0$

Line 1 selects row x and column y, which have the optimal similarity value. Line 2 changes all of the values in line x to 0 so that they are not re-counted in the next iteration. Moreover, Line 3 changes all of the values in line y to 0 so that they are not re-counted in the next iteration.

The next element of similarity is the message similarity ($msSim$) of sequence diagrams d_1 and d_2. The similarity of the messages can be seen in Equation 3.

$$msSim(d_1, d_2) = \frac{2 \times (\sum_{k=1}^{\min(|MS_1|,|MS_2|)} changePivot(\sum_{i=1}^{|MS_1|} \sum_{j=1}^{|MS_2|} dmsSim(ms_i, ms_j)))}{|MS_1| + |MS_2|} \tag{3}$$

Equation 3 also implements the greedy algorithm to semantically assess the message collection (MS_1) in sequence diagram d_1 to the message collection (MS_2) in sequence diagram d_2. Equation 3 also uses *changePivot*, which has been described in Algorithm 1. Message ms_i is a message from d_1 at index i. Message ms_j is a message from d_2 at the index to j. Messages ms_i and ms_j cannot be assessed directly because a message consists of several pieces of information (as shown in Figure 1); therefore, a detailed message similarity assessment ($dmsSim$) is required (which is presented in Equation 4).

$$\begin{aligned} dmsSim(ms_1, ms_2) = & w_{cs} \times cosineSim(cSrc_1, cSrc_2) + w_t \times Sim(t_1, t_2) \\ & + w_{mn} \times cosineSim(mn_1, mn_2) \\ & + w_{cd} \times cosineSim(cDst_1, cDst_2) \end{aligned} \tag{4}$$

A detailed message similarity assessment ($dmsSim$) consists of four parts. Each part has its own weight, where the total of all weights is 1. First, the similarity of the source class ($cSrc$) of the two messages with weight w_{cs} was assessed by cosine similarity. Second, the similarity of the type (t) of the two messages with weight w_t was assessed by 0 or 1. If the message types are the same, then the similarity value is 1. If the message type is different, then the similarity value is 0. Third, the similarity of the message name (mn) of the two messages with weight w_{mn} was assessed by cosine similarity. Last, the similarity of the destination class ($cDst$) of the two messages with weight w_{cd} was assessed based on cosine similarity.

3. Subgraph Edit Distance

The structural assessment of sequence diagrams was performed using subgraphs that were derived from graphs translated from class diagrams. The graph resulting from class diagram translation is known as a UML common graph (UCG). The subgraph in the first diagram was then compared with the subgraph in the second diagram using graph edit distance [24, 26]. Therefore, we call the structural assessment approach 'subgraph edit distance.' Notably, our structural approach ignores label information from sequence diagrams. Structural assessment only uses information on the form of the components owned and the message passing flow.

3.1. UML Common Graph Translation

The UCG was inspired by the UML class graph proposed by Yuan [33]. The purpose of a UCG is to create a graph model that can represent more than one type of UML diagram. We implemented this model in class diagrams and sequence diagrams. Notably, a UCG is a directed graph. The direction of a UCG is obtained from the direction of the message passing between class objects. A UCG consists of V and E, where V is a collection of vertices, and E is a collection of edges between two vertices. Table 1 describes the types of vertices and edges of the UCG model.

Based on Table 1, we added one vertex and four edges from the previous research [33]. The addition of the UCG component also intends to represent sequence diagrams. The shape of the UCG translated from the sequence diagram is the same as the form of the UCG from the class diagram (as explained in previous studies [33]). For example, Figure 2 is the translation of a sequence diagram into a UCG.

The sequence diagram in Figure 2 consists of two classes: $Class_1$, and $Class_2$. $Class_1$ translates to vc_1; also, class $Class_1$ has an object called $Object_1$. $Object_1$ translates to vob_1. Operation $operation_{11}$ is an operation that is called from $Class_1$.

Table 1
UML Common Graph Description

No.	Element Type	Name	*Tag*
1.	Vertex	Class Vertex	vc
2.	Vertex	Attribute Vertex	va
3.	Vertex	Operation Vertex	vo
4.	Vertex	Parameter Vertex	vp
5.	Vertex	Object Vertex	vob
6.	Edge	Attribute Edge	ea
7.	Edge	Operation Edge	eo
8.	Edge	Parameter Edge	ep
9.	Edge	Association Edge	e_1
10.	Edge	Generalization Edge	e_2
11.	Edge	Aggregation Edge	e_3
12.	Edge	Composition Edge	e_4
13.	Edge	Dependency Edge	e_5
14.	Edge	Realization Edge	e_6
15.	Edge	Class Edge	ec
16.	Edge	Reply Message Edge	e_7
17.	Edge	Synchronous Message Edge	e_8
18.	Edge	Asynchronous Message Edge	e_9

Operation $operation_{11}$ translates to vo_{11}. Class $Class_2$ translates to vc_2. Class $Class_2$ has $Object_2$ and $Object_3$ as well as vob_2 and vob_3, respectively. $Class_2$ has two operations used in the sequence diagram: $operation_{21}$, and $operation_{22}$. $operation_{21}$ translates to vo_{21}. Operation $operation_{21}$ has a parameter (par_{21}), which translates to vp_{211}. Operation $operation_{22}$ translates to vo_{22}. The relationship between the class and the translation operations becomes eo. The relationship between the class and object of transformation becomes eob. The relationship between the operations and the parameters translates to ep. The vc_1 and vc_2 classes are linked to three messages of the same type (known as synchronous messages). The first message starts from $Class_1$ to $Class_2$. This relationship translates to an edge leading from vc_1 to vc_2 with the synchronous message type (e_8). The second message starts from $Class_2$ to $Class_1$. This relationship translates to an edge leading from vc_2 to vc_1 with the synchronous message type (e_8). The third message starts from $Class_1$ to $Class_2$ with the same synchronous message type (e_8). Since the third message is the same as the first, we do not need to make a new edge anymore.

3.2. Structural Assessment

The assessment in this section strictly focuses on the structures and types of graph vertices and edges. The label information from each vertex and edge is ignored. As per previous studies [33], the similarity assessment between two UCGs was divided

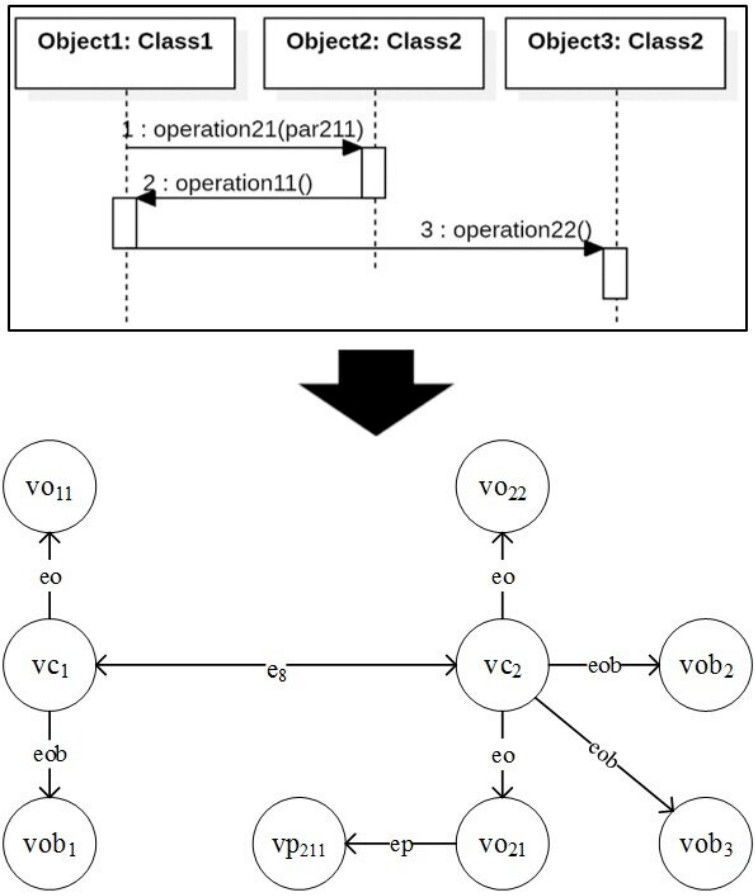

Figure 2. Translation sequence diagram into UCG

into two portions: *intraSim*, and *interSim*. The relationship between *intraSim* and *interSim* can be observed in Equation 5.

$$strucSim(g_1, g_2) = ((1 - \rho_{str}) \times intraSim(g_1, g_2)) + (\rho_{str} \times interSim(g_1, g_2)) \quad (5)$$

Equation 5 assesses the similarity between UCG g_1 and UCG g_2 ($strucSim$). The value of ρ_{str} explains the importance of *intraSim* and *interSim*. The value of ρ ranges from 0 to 1. The *intraSim* is a structural assessment between the classes from the first UCG with the classes from the second UCG. The information taken from a class in *intraSim* includes the owned object, the operation called from that class, and the accompanying parameters. Based on Figure 2, the information used to assess *intraSim* includes the two generated subgraphs. Both subgraphs are presented

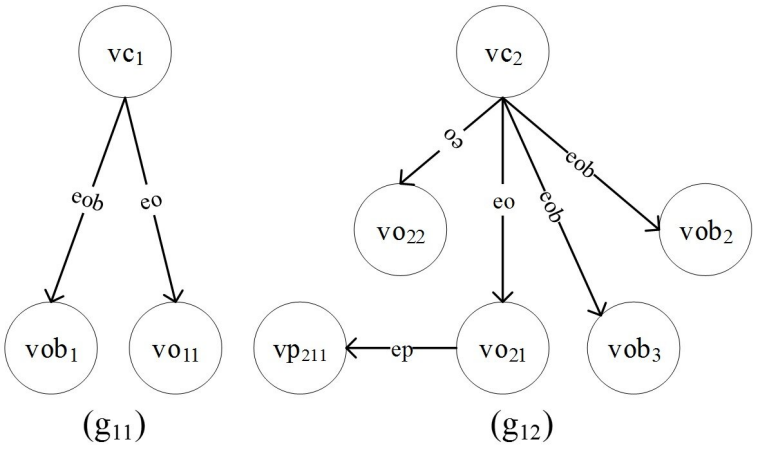

Figure 3. Subgraphs in *intraSim* assessment

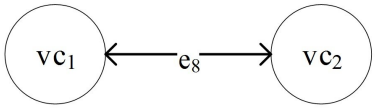

Figure 4. Subgraphs in *interSim* assessment

in Figure 3. Graph g_{11} and g_{12} are the subgraphs that will be used from one UCG as a comparison to the subgraphs from other UCGs to assess *intraSim*. Equation 6 presents *intraSim* assessment.

$$intraSim(g_1, g_2) = \frac{2 \times (\sum_{k=1}^{\min(|SG_1|,|SG_2|)} changePivot(\sum_{i=1}^{|SG_1|} \sum_{j=1}^{|SG_2|} GED(sg_i, sg_j)))}{|SG_1| + |SG_2|} \tag{6}$$

intraSim also uses a greedy algorithm because it searches for the optimal value of first set of UCG subgraphs SG_1 with second set of UCG subgraphs SG_2. The similarity assessment between the two subgraphs (sg_i and sg_j) was assessed using graph edit distance (GED) [9, 24, 25].

interSim is the structural similarity between the relationships between the classes from the first UCG and the relationships between the classes from the second UCG. The information used in *interSim* is the relationship between the classes held by UCG. Each UCG has only one subgraph that describes *interSim*. Figure 4 displays the subgraph that was generated from UCG in Figure 2. Since each UCG has only one subgraph for assessing *interSim*, the formula for evaluating *interSim* is simpler than that for *intraSim*. Equation 7 presents the formula for assessing

interSim. The *interSim* of g_1 and g_2 can be directly calculated using GED based on the subgraphs that are generated from each compared UCG.

$$interSim(g_1, g_2) = GED(sg_1, sg_2) \tag{7}$$

4. Sequence Diagram Assessment

Sequence diagram assessment ($seqSim$) is a combination of semantic assessment ($semSim$) using label similarity and structural assessment ($strucSim$) using subgraph edit distance. Notably, semantic and structural assessments have different levels of importance. This difference is indicated by the value of ρ in Equation 8. The value of ρ ranges from 0 to 1. The compared sequence diagrams are d_1 and d_2. Additionally, the results of the translational diagrams compared into a UCG are g_1 and g_2.

$$seqSim(d_1, d_2) = ((1 - \rho) \times semSim(d_1, d_2)) + (\rho \times strucSim(g_1, g_2)) \tag{8}$$

5. Experiment

The experiment was conducted in two stages. The first stage involved deciding the gold standard based on the responses of experts. The second stage involved testing the reliability of the assessment method. The reliability of the proposed method was analyzed by Gwet's AC1 [15, 18, 32]. In evaluating reliability among experts, Gwet's AC1 is better than Kappa Statistics. The representation of agreement values [20] is presented in Table 2; agreement ranges from 'less than chance agreement' to 'almost perfect agreement.'

Table 2
Agreement Interpretation

Index Value	Proportion of Agreement
<0	Less than chance agreement
0.01 − 0.20	Slight agreement
0.21 − 0.40	Fair agreement
0.41 − 0.60	Moderate agreement
0.61 − 0.80	Substantial agreement
0.81 − 1	Almost perfect agreement

The sequence diagram dataset was collected by providing three questions to create a sequence diagram based on one use case description given to students. Each question was answered by at least eight students. Each student created only one sequence diagram (for a total of 28 sequence diagrams). A summary of the dataset is provided in Table 3.

Table 3
Dataset summary

Project	Description	Use Case Name	Number of Answers	Average Object	Average Message
Outlay	Application that records user's financial history.	Record expenditure	10	5	13
Library	Book rental application.	Borrow book	9	5	11
QuickBill	Point of sale application.	Add new transaction	9	4	13

5.1. Gold Standard

A gold standard was created to calculate the suitability of the expert answers with the results of the proposed method. Based on the three questions given to the students, we created an answer key in the form of sequence diagrams. We then created a questionnaire to assess the similarity between the answer keys for the answers of the students. The respondents included 23 experts; these experts were lecturers who teach software engineering, individually assess software design using UML diagrams, and have taught UML diagrams for at least two years. The respondents assessed the students' answers based on the provided answer keys. The respondents' answers were then tested for validity and reliability. These answers subsequently became the gold standard for testing the reliability of this method.

5.2. Result

We compared all combinations of ρ, ρ_{sem}, and ρ_{str} with the averaged expert answers. The combined value of ρ, ρ_{sem}, and ρ_{str} consisted of 0, 0.1, 0.2, to 1. A total of 1,331 combinations were performed. The results of our proposed method and standard answers were converted to ordinal numbers (from 1 to 5) with a balanced distribution of ranges. The aim was to facilitate the assessment of the agreement between the proposed method and the experts using Gwet's AC1. The highest value of Gwet's AC1 for each ρ is presented in Figure 5. Figure 5 shows that the highest agreement value was 0.728 at a ρ of 0.8. The values of ρ_{sem} were 0.9 and 1. The results for ρ_{str} were 0.2 and 0.3.

6. Discussion

This study supports Triandini's idea [31] that semantic and structural aspects are relevant aspects for assessing the similarity of sequence diagrams. Figure 5 shows that the value of the agreement is 0.728. Based on Table 2, this value can be interpreted as a substantial agreement between the proposed approach and the expert responses.

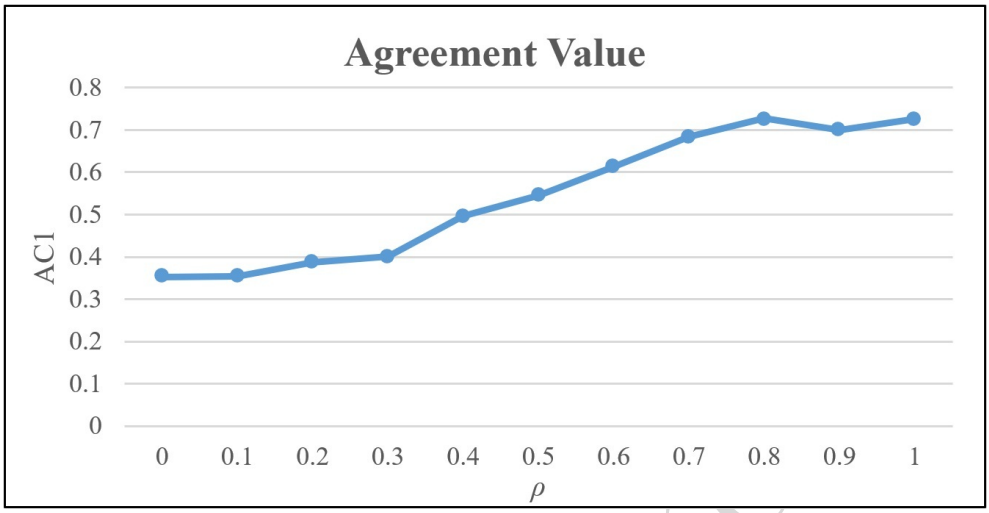

Figure 5. Highest agreement value of each ρ

Therefore, our proposed method can work as reliably as experts in assessing the similarity of two sequence diagrams.

The data we used as the gold standard originated from 28 pairs of diagrams with satisfactory reliability (Cronbach's alpha of 0.979). However, the agreements between experts were also substantial. The value for inter-rater reliability evaluation is only 0.62. If the number of respondents and the agreement among experts were to increase, the reliability of our proposed method would also be expected to increase proportionally.

Figure 5 shows that the best ρ value is 0.8. Overall, the experts observed a structural assessment of 80%, while the observed semantic assessment was only 20%. This suggests that experts tend to consider structural assessment rather than semantic assessment. This is also supported by an increase in the curve from ρ 0 to 1. At ρ 0.8, ρ_{sem} is 0.9 and 1. This shows that the experts nearly ignored the label information from the objects in the sequence diagrams; instead, the experts only looked at the label information of the message and the object order that connected the message. This argument is based on the average number of objects and messages in Table 3 and ρ_{sem} that are found. Thus, it can be concluded that the number of components is directly proportional to the tendency of experts to semantically assess. Table 3 shows that the number of messages is greater than the number of objects. Also, ρ_{str} is 0.2 and 0.3 at ρ 0.8. This suggests that the experts tended to view the intra-structure of the object class rather than the inter-structure. This is in line with semantic similarity (which only considers the order of messages).

In the present study, we separately analyzed semantic assessment and structural assessment. For the semantic assessment, the value generated at ρ_{sem} 0.1 is lower

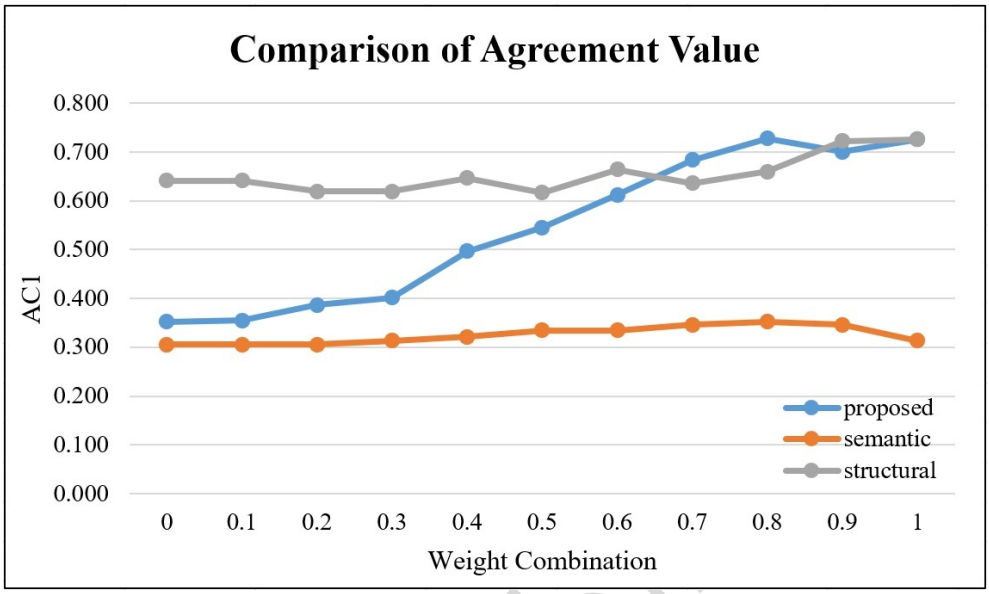

Figure 6. Comparison of agreement value

than the average expert response. This suggests that the label understanding of the proposed method is still not equivalent to that of the experts. On the other hand, the structural assessment value generated at ρ_{str} 0.5 is greater than the average expert response. This is because the proposed UCG is calculated by using GED. GED calculates graph similarity more accurately than experts. This especially applies to complicated graph shapes. As shown in Table 3, the number of messages in each diagram is greater than the number of objects. Therefore, it is evident that some experts are not very accurate in assessing large structures.

Further comparisons were made between the proposed method, semantic assessment, and structural assessment. Figure 6 presents the results of our comparison. The y-axis is the agreement value of Gwet's AC1, and the x-axis is the weight combination of the assessments. ρ represents the weight combination of our proposed method based on Equation 8, ρ_{sem} represents the weight combination of semantic assessment based on Equation 1, and ρ_{str} represents the weight combination of structural assessment based on Equation 5. The highest agreement value of the semantic assessment is 0.352 at a weight of 0.8. The highest agreement value of the structural assessment is 0.726 at a weight of 1. Therefore, our proposed method has the highest agreement value – it is 0.728 at a weight of 0.8.

Our findings indicate that the experts largely assessed the similarity of sequence diagrams by observing the structural similarity between two diagrams. This is in line with a sequence diagram, which shows the flow of a process. Sequence diagrams explain how the order of objects communicate with each other through messages.

We believe that, although the number of respondents and the agreement of experts increased, a higher weight remains in structural similarity. Moreover, the semantic similarity weight is also the same. Experts will tend to observe the lexical of a message based on each object rather than a lexical collection that contains the source and destination information of the message. However, it remains unknown whether the weight of structural similarity will change or not. Our findings show that *intraSim* and *interSim* are still considered equal. Experts will likely tend to look at *interSim* rather than *intraSim*, as *interSim* can be easily understood.

The method we propose can help experts make assessments based on their needs. This is because we separate the components in a diagram. Thus, experts can adjust the weight of each component based on their needs. Therefore, our proposed method not only assesses similarity but can also be used for reusing and clone detection. The weight settings for reusing and clone detection may be evenly distributed across each component.

7. Conclusion

We have presented a novel approach for automated behavioral diagram assessment, which can be especially useful for sequence diagrams. The assessment is divided into two approaches: the semantic approach (using label similarity), and the structural approach (using subgraph edit distance). Notably, the proposed approach for assessing the similarity of two sequence diagrams can be as reliable as an expert's assessment. The agreement between the proposed approach and the experts' assessment reached a substantial agreement. When assessing the similarity of sequence diagrams, experts tend to observe structural aspects rather than semantic aspects. This is in line with the purpose of the sequence diagram, which is to describe the behavior in a use case by describing the structure of the message flow.

This research can be developed further through assessments using a combination of semantic and structural approaches implemented in other UML diagrams. Furthermore, the assessment of similarity in sequence diagrams cannot only be assessed by similar diagrams. Sequence diagrams should be similar to the other diagrams (e.g., class diagrams).

Acknowledgements

This research was funded by the Ministry of Research and Technology/National Agency for Research and Innovation of the Republic of Indonesia. This research is a collaboration among Institut Teknologi Sepuluh Nopember, Politeknik Negeri Banjarmasin, and Institut Teknologi Dan Bisnis STIKOM Bali.

References

[1] Adamu A., Zainon W.: A review of UML model retrieval approaches. *Indian Journal of Science and Technology*, vol. 9(46), pp. 1–8, 2016.

[2] Adamu A., Zainon W.M.N.W.: Matching and retrieval of state machine diagrams from software repositories using Cuckoo Search Algorithm. In: *2017 8th International Conference on Information Technology (ICIT)*, pp. 187–192. IEEE, 2017.

[3] Adamu A., Zainon W.M.N.W.: Multiview Similarity Assessment Technique of UML Diagrams. *Procedia Computer Science*, vol. 124, pp. 311–318, 2017.

[4] Adamu A., Zainon W.M.N.W.: Similarity Assessment of UML Sequence Diagrams Using Dynamic Programming. In: *International Visual Informatics Conference*, pp. 270–278. Springer, 2017.

[5] Bloxham S., den Outer B., Hudson J., Price M.: Let's stop the pretence of consistent marking: exploring the multiple limitations of assessment criteria. *Assessment & Evaluation in Higher Education*, vol. 41(3), pp. 466–481, 2016.

[6] Buijs S., Heerkens J., Ampe B., Delezie E., Rodenburg T., Tuyttens F.: Assessing keel bone damage in laying hens by palpation: effects of assessor experience on accuracy, inter-rater agreement and intra-rater consistency. *Poultry science*, vol. 98(2), pp. 514–521, 2019.

[7] Castro L.J.G., Berlanga R., Garcia A.: In the pursuit of a semantic similarity metric based on UMLS annotations for articles in PubMed Central Open Access. *Journal of Biomedical Informatics*, vol. 57, pp. 204–218, 2015.

[8] Chonoles M.J.: *OCUP 2 Certification Guide: Preparing for the OMG Certified UML 2.5 Professional 2 Foundation Exam.* Morgan Kaufmann, 2017.

[9] Daller É., Bougleux S., Gaüzère B., Brun L.: Approximate graph edit distance by several local searches in parallel. In: *7th International Conference on Pattern Recognition Applications and Methods*. 2018.

[10] Fauzan R., Siahaan D., Rochimah S., Triandini E.: Class Diagram Similarity Measurement: A Different Approach. In: *2018 3rd International Conference on Information Technology, Information System and Electrical Engineering (ICITISEE)*, pp. 215–219. IEEE, 2018.

[11] Fauzan R., Siahaan D., Rochimah S., Triandini E.: A Different Approach on Automated Use Case Diagram Semantic Assessment. *International Journal of Intelligent Engineering and Systems*, vol. 14(1), pp. 496–505, 2021.

[12] Fellbaum C.: WordNet. In: *Theory and applications of ontology: computer applications*, pp. 231–243. Springer, 2010.

[13] Feng Y., Bagheri E., Ensan F., Jovanovic J.: The state of the art in semantic relatedness: a framework for comparison. *The Knowledge Engineering Review*, vol. 32, 2017.

[14] Fischer A., Riesen K., Bunke H.: Improved quadratic time approximation of graph edit distance by combining Hausdorff matching and greedy assignment. *Pattern Recognition Letters*, vol. 87, pp. 55–62, 2017.

[15] Gwet K.: Kappa statistic is not satisfactory for assessing the extent of agreement between raters. Series: Statistical Methods for Inter-Rater Reliability Assessment 1 (1): 1–5. *Gaithersburg: STATAXIS Consulting*, vol. 4, 2002.

[16] Harispe S., Ranwez S., Janaqi S., Montmain J.: Semantic similarity from natural language and ontology analysis. *Synthesis Lectures on Human Language Technologies*, vol. 8(1), pp. 1–254, 2015.

[17] Jenkins D., Simpson S., Peacock A.: Investigating the consistency and quality of EPC ratings and assessments. *Energy*, vol. 138, pp. 480–489, 2017.

[18] Jimenez A.M., Zepeda S.J.: A Comparison of Gwet's AC1 and kappa when calculating inter-rater reliability coefficients in a teacher evaluation context. *Journal of Education Human Resources*, p. e20190001, 2020.

[19] Kutuzov A., Dorgham M., Oliynyk O., Biemann C., Panchenko A.: Learning Graph Embeddings from WordNet-based Similarity Measures. *arXiv preprint arXiv:1808.05611*, 2018.

[20] Landis J.R., Koch G.G.: The measurement of observer agreement for categorical data. *biometrics*, pp. 159–174, 1977.

[21] Majumder G., Pakray P., Gelbukh A., Pinto D.: Semantic textual similarity methods, tools, and applications: A survey. *Computación y Sistemas*, vol. 20(4), pp. 647–665, 2016.

[22] Park W.J., Bae D.H.: A two-stage framework for UML specification matching. *Information and Software Technology*, vol. 53(3), pp. 230–244, 2011.

[23] Pressman R.S.: *Software engineering: a practitioner's approach*. Palgrave macmillan, 2005.

[24] Riesen K., Bunke H.: Graph Edit Distance Novel Approximation Algorithms. In: *Handbook of Pattern Recognition and Computer Vision*, pp. 275–291. World Scientific, 2016.

[25] Riesen K., Ferrer M., Bunke H.: Approximate graph edit distance in quadratic time. *IEEE/ACM transactions on computational biology and bioinformatics*, vol. 17(2), pp. 483–494, 2015.

[26] Riesen K., Ferrer M., Dornberger R., Bunke H.: Greedy graph edit distance. In: *International Workshop on Machine Learning and Data Mining in Pattern Recognition*, pp. 3–16. Springer, 2015.

[27] Robinson W.N., Woo H.G.: Finding reusable UML sequence diagrams automatically. *IEEE software*, vol. 21(5), pp. 60–67, 2004.

[28] Salami H.O., Ahmed M.: Retrieving sequence diagrams using genetic algorithm. In: *2014 11th International Joint Conference on Computer Science and Software Engineering (JCSSE)*, pp. 324–330. IEEE, 2014.

[29] Siahaan D., Desnelita Y., et al.: Structural and semantic similarity measurement of UML sequence diagrams. In: *2017 11th International Conference on Information & Communication Technology and System (ICTS)*, pp. 227–234. IEEE, 2017.

[30] Sommerville I.: Software engineering 9th Edition. *ISBN-10*, vol. 137035152, p. 18, 2011.

[31] Triandini E., Fauzan R., Siahaan D.O., Rochimah S.: Sequence Diagram Similarity Measurement: A Different Approach. In: *2019 16th International Joint Conference on Computer Science and Software Engineering (JCSSE)*, pp. 348–351. IEEE, 2019.

[32] Wongpakaran N., Wongpakaran T., Wedding D., Gwet K.L.: A comparison of Cohen's Kappa and Gwet's AC1 when calculating inter-rater reliability coefficients: a study conducted with personality disorder samples. *BMC medical research methodology*, vol. 13(1), p. 61, 2013.

[33] Yuan Z., Yan L., Ma Z.: Structural similarity measure between UML class diagrams based on UCG. *Requirements Engineering*, pp. 1–17, 2019.

Affiliations

Reza Fauzan
Department of Informatics, Institut Teknologi Sepuluh Nopember, Surabaya, Indonesia, Department of Electrical Engineering, Politeknik Negeri Banjarmasin, Banjarmasin, Indonesia, reza.fauzan@poliban.ac.id

Daniel Siahaan
Department of Informatics, Institut Teknologi Sepuluh Nopember, Surabaya, Indonesia, daniel@if.its.ac.id

Siti Rochimah
Department of Informatics, Institut Teknologi Sepuluh Nopember, Surabaya, Indonesia, siti@if.its.ac.id

Evi Triandini
Information Systems, Institut Teknologi dan Bisnis STIKOM Bali, Denpasar, Indonesia, evi@stikom-bali.ac.id

Received: 20.06.2020
Revised: 25.11.2020
Accepted: 29.12.2020