2023 IEEE International Conference on Advanced Learning Technologies (ICALT) | 979-8-3503-0054-3/23/$31.00 ©2023 IEEE | DOI: 10.1109/ICALT58122.2023.00071

A machine learning approach for the identification of learners' misconceptions in algebraic problem-solving

Joice Cazanoski Gomes
Graduate Program in Informatics (PPGInf)
Universidade Federal do Paraná (UFPR)
Curitiba, Brazil
ORCID: 0000-0002-5873-1426

Patricia A. Jaques
Graduate Program in Informatics (PPGInf)
Universidade Federal do Paraná (UFPR)
Graduate Program in Computing (PPGC)
Universidade Federal de Pelotas (UFPEL)
Curitiba and Pelotas, Brazil
ORCID: 0000-0002-2933-1052

***Abstract*—Misconceptions play a significant role in the learning process as they reflect an inaccurate understanding of a particular concept. Error diagnosis can help teachers and intelligent learning environments determine the most appropriate type of student assistance. Previously, misconceptions were identified using rule-based expert systems (bug libraries) and clustering algorithms. Bug libraries demand extensive work from developers to identify all potential misconceptions and code rules for each one in advance. Additionally, these solutions cannot detect misconceptions for which rules were not explicitly programmed. Clustering-based solutions overcome these drawbacks by automatically identifying misconceptions based on students' most common errors. To effectively and efficiently identify misconceptions, clustering solutions must have a suitable representation of the problem and its steps, and employ machine learning algorithms capable of discerning patterns from them. This paper proposes a solution that utilizes expression trees to represent algebraic problem-solving steps and the Density-Based Spatial Clustering of Applications with Noise (DBSCAN) algorithm to identify misconceptions by clustering similar errors in a database containing 1064 steps from 112 students. This database was collected from an intelligent learning system designed to assist in solving first-degree equations. In our final solution, a Natural Language Processing tokenizer was employed to represent each term numerically, which identified 178 homogeneous clusters with minimal noise and few outliers.**

***Index Terms*—misconceptions, clustering, smart learning environments, algebra learning**

I. Introduction

A misconception is typically described as a mistake brought on by misinterpreting a concept. According to Schmidt [1], misconceptions are errors committed by students due to poor or absent understanding of a particular concept required to solve a problem. It is different from an error, which the student can identify by herself when she makes it. Consequently, misconceptions can hinder a student's ability to understand new concepts and can be difficult for teachers to identify and address.

Misconceptions are unavoidable in mathematics and can be interpreted as an alternative method for determining the correct answer. A math misconception has two unique characteristics: error repetition and independence from numerical values. For example, let's consider that the equation $x + 4 = 10$ was presented for a student to solve, and she made a mistake, passing the value from left to right with the wrong signal ($x = 10+4$). If the student knows the right concept and understands it is necessary to invert the operation, she can identify her error and correct it. On the other hand, if this student does not know this concept, she will probably repeat the same mistake many times, which is considered a misconception.

A deeply rooted misconception can create obstacles and make the learning process more challenging for students [2]. Consequently, identifying misconceptions is necessary for a teacher or learning environment to provide adequate individualized assistance to students.

Although the automatic detection of misconceptions is still considered a challenging task, it is directly connected to the capacity of the artificial tutor to improve the student's knowledge in a specific domain [3]. By utilizing automated techniques to identify misconceptions, educators and artificial tutors can gain valuable insights into students' learning gaps, allowing them to adjust their teaching strategies accordingly. The early detection and correction of misconceptions can lead to more focused and effective instruction, ultimately resulting in improved academic performance [4], [5]. Additionally, the automatic identification of misconceptions can enable personalized learning experiences, as students receive tailored feedback and support based on their individual needs.

Since 2014, the number of researchers attempting to identify mathematical misconceptions has increased [2]. However, most research has focused on developing and applying cognitive diagnosis models (CDM), also known as a bug library. These techniques efficiently identify misconceptions within the

This study was financed in part by the following research funding agencies: CAPES (Finance Code 001), FAPERGS (granting 17/2551-0001203-8), and CNPq (granting 306005/2020-4) from Brazil.

2161-377X/23/$31.00 ©2023 IEEE
DOI 10.1109/ICALT58122.2023.00071

allotted time that assessments are administered. Bug libraries require considerable effort from developers to identify all possible misconceptions and create rules for each one beforehand. Furthermore, these approaches are incapable of identifying misconceptions that lack explicitly designed rules. Clustering-based solutions address these drawbacks by automatically identifying misconceptions based on students' most common errors. However, this approach requires a database of students' error logs for the machine learning method to learn from, as well as an appropriate method for representing the input to automatically detect misconceptions.

This article proposes a clustering-based solution to automatically identify misconceptions in algebraic problem-solving. Our approach uses the DBSCAN (Density-Based Spatial Clustering of Applications with Noise) algorithm to cluster similar classes of errors from the tutor students' log that could indicate a misconception and expression trees to represent the steps, i.e., the equations that compose each step of a learner's solution. Knowing and understanding the structure of an equation, not only its result, is necessary to identify an equation misconception. Expression trees allow for a structured representation of an equation and clarify its order of operations. Consequently, this type of representation eliminates the need for parentheses. We also investigate the use of Natural Language Processing as a pre-processing step in identifying patterns of misconceptions.

To evaluate our approach, we used log data from the algebraic intelligent tutoring system PAT2Math (Personal Affective Tutor to Math). This system provides step-based tutoring that helps students solve first-degree algebraic equations [6]. We collected 1064 incorrect steps from 112 students who solved a total of 330 first-degree equations.

II. Related work

Feldman et al. [7] proposed a method to identify unknown misconceptions by analyzing students' answers and all possible paths to obtain such answers. They collected data from 296 students using the MetaMetrics learning system, consisting of answers to 32 addition and subtraction problems. The study defined solution groups with three or more solutions from the same student, resulting in 868 groups, of which 111 had at least one misconception. The algorithm was able to reconstruct the solution for 86% of the groups, with 77 of these being classified as accurate or partially accurate. The authors mentioned a limitation of the MetaMetrics system, which only allows the final solution of a problem to be entered, leaving out intermediate steps that could be useful in identifying misconceptions. As a result, the proposed solution assumes that all problems within a group are solved in a similar manner.

Andersson et al. [8] developed a solution to identify misconceptions in a math learning environment for children aged 6 to 10 by using the fuzzy variant of the DBSCAN algorithm (FN-DBSCAN) that assigns a confidence level to each error linked with a cluster. The algorithm was evaluated with simulated users having known misconceptions, and after twenty and forty iterations, it identified 107 errors and one possible misconception, and 206 errors and three possible misconceptions, respectively. The authors concluded that the algorithm performs better in the long-term analysis of user data.

Elmadani [3] proposed using data mining to identify misconceptions in the EER-Tutor, an Intelligent Tutoring System (ITS) used to teach database conceptual model creation and characterized as a constraint-based ITS. Despite providing feedback for timing errors, the EER-Tutor cannot determine if these errors are due to misconceptions. The authors used the FP-Growth algorithm to identify potential misconceptions based on the most common relationships between incorrect answers in multiple-choice questions. The algorithm processed 1135 answers, generating 912 sets with 238 misconceptions identified. The authors developed a results-based hierarchy, and experts analyzed the results, considering them satisfactory. As enhancements, the authors proposed recognizing that different restrictions may share a similar structure and connect to the same misconception.

Gomes et al. [9] present a closely related study, which also employs clustering algorithms to identify algebraic misconceptions in a step-based tutoring system that assists students in solving equations step by step. In this work, the equations were normalized and clustered using k-modes. The tests only retrieved records in which the student entered an incorrect answer. The data consist of first-degree equation-solving steps and minimal feedback from the tutor (whether a step was right or wrong). The results indicated that it is possible to identify misconceptions automatically using clustering. However, the structure used to represent the equation and the large number of special terms in an equation (e.g., parentheses) led to some issues and incorrect results.

The main difference between our work and related studies lies in the type of input, its representation, and the methods used to identify misconceptions. While Feldman et al. [7] only considered students' final solutions to identify misconceptions, our approach utilizes expression trees to represent all the intermediate algebraic problem-solving steps and employs the DBSCAN algorithm to cluster similar errors. In contrast to Andersson et al. [8], our study is based on real data, not simulated data. Furthermore, unlike Elmadani's approach [3], our method investigates problem-solving steps themselves and is not limited to analyzing multiple-choice questions. Lastly, we addressed issues encountered by Gomes et al. [9] in representing equations by using expression trees and a Natural Language Processing tokenizer to represent each term numerically, resulting in more accurate clustering and misconception identification.

III. Method

This paper proposes using the DBSCAN clustering algorithm to identify algebraic misconceptions from a step-based tutoring system's database log of student responses. We also use the expression tree to represent the problem-solving steps of the students.

An expression tree is a way to represent mathematical expressions in a hierarchical, tree-like structure. Each internal node of the tree represents an operator, and each leaf node represents an operand. Using an expression tree to represent a mathematical expression has several advantages. It allows easy evaluation of the expression by traversing the tree and performing the operations in a natural order. Moreover, it allows for easy modification of the equation, such as changing the order of operations or adding or removing terms. And it permits easy expression simplification by applying mathematical identities and rules.

Furthermore, using expression trees to represent mathematical expressions clarifies the order of operations [10]. Consequently, it takes care of precedence and associativity and eliminates the need for parentheses. Prior research has indicated that the representation of mathematical expression in problem-solving can have an effect on the outcomes of machine learning algorithms [9].

As part of the pre-processing step, we use a tokenizer to split the equation into each operator and operand. We used the Natural Language Toolkit (NLTK), a popular Python library, to work with human language data. It provides various tools for tokenizing, parsing, and analyzing text, including the NLTK tokenizer[1]. The NLTK tokenizer is a suite of tokenization functions that can split the text into smaller units, such as words or sentences. These smaller units, known as tokens, can then be processed and analyzed to gain insights into the text's meaning and structure [11].

Using this tokenizer, we map each part to a number, creating a unique numerical sequence for each equation structure instead of using a textual representation. We made this decision to become easier to process these entries with clustering algorithms because they work much better with numerical groups.

Once we had the equations represented by expression trees and tokenized, we applied the DBSCAN. The DBSCAN algorithm is a density-based clustering method used to identify clusters of points in a dataset. It defines clusters as areas of high density and separates clusters from one another based on low-density areas. The algorithm has two main parameters: the radius (eps) and the minimum number of points ($minPts$) required to form a dense region. The algorithm starts by selecting a random point and then identifies all other points within the defined radius from it. A new cluster is formed if there are at least $minPts$ points within that radius. The algorithm then repeats this process for each point in the new cluster and continues to expand the cluster until no more points within the radius can be found. Points that are not part of any cluster are considered noise. DBSCAN is particularly useful for identifying clusters of arbitrary shape and for handling clusters with varying densities.

A. Data extraction

The data used in the model was extracted from the PAT2Math databases. PAT2Math is a step-based intelligent tutoring system (ITS) to assist students while solving first-degree equations, i.e., it provides assistance for each step of the learner problem-solving in the form of minimal feedback (right or wrong), feedback error, and also hints when the learner is blocked and do not know how to proceed. To be able to provide assistance, the ITS uses a domain model that is implemented as an expert system [12]. In PAT2Math, the cognitive model, which is an extension of the domain model, is responsible for assessing and correcting, if necessary, each step submitted by the learner. The equations for the students to solve are organized in levels of difficulty. Each level contains plans with isomorphic equations that works the same algebraic operations. The student should solve a minimum number of equations in a plan that shows that he or she masters the algebraic operations worked to access the next plan.

For each step of the student, the PAT2Math database stores a record. Each step's register has the student id, the equation level of difficulty, the current step provided by the learner, his or her previous step, and the minimal feedback provided by PAT2Math for the step (whether it is right or wrong). We must first notice and evaluate the error the student made to identify a misconception. Consequently, we only need to retrieve the registers where the current step is wrong. So, in this study, we use the wrong and previous steps.

The dataset we used contained 1,319 entries, which we divided into training and testing sets. We used 1,064 entries for training and 255 entries for testing. This data was obtained from a subset of the PAT2Math database, which originally included 1,319 steps from 112 students who collectively solved 330 first-degree equations.

B. Pre-processing

We divided the pre-processing into three steps: 1) equation standardization, 2) expression tree creation with postfix notation, and 3) applying the NLTK tokenizer. Table I illustrates the data transformation steps for the equation $x + 5 = 7$.

First (**Step 1**), we standardized the equation by replacing numerical values with letters. Algebraic misconceptions do not depend on the numeric values of the terms in an equation. Therefore, equations such as $x+3 = 6$ and $x+2 = 4$ represent isomorphic equations, which can be denoted by $x + a = b$. This allows for the identification of misconceptions because math expressions, like equations, have a structured form that is different from textual responses.

Second (**Step 2**), we represented the equation as an expression tree using postfix notation, also known as "Reverse Polish notation". This is a mathematical notation in which operators follow their operands, and it does not require parentheses like infix notations, making it simpler for machine learning algorithms to use.

In the third step (**Step 3**), we utilized the NLTK tokenizer. The NLTK tokenizer is responsible for transforming algebraic expressions into a numerical representation by replacing each character with a number from a created alphabet mapping. Therefore, the equation $x + a = b$ is represented by 15723 in numerical form. This numerical representation enables more

[1] Available at https://github.com/nltk/nltk/wiki/Articles-about-NLTK

efficient processing and analysis when identifying misconceptions using the DBSCAN clustering algorithm.

	Step	Input	Output
1	Standardization	x + 5 = 7	x + a = b
2	Expression Tree (Postfix)	x + a = b	x a b + =
3	NLTK Tokenizer	x a b + =	e.g.: 1, 5, 7, 2, 3

TABLE I
DATA TRANSFORMATION

C. Clustering

We applied the Density-based spatial clustering of applications with noise (DBSCAN). Unlike k-means/k-modes, DBSCAN does not require the number of clusters to be provided as a parameter, but it does require the following parameters to be set:

- *eps*: The *eps* value represents the maximum distance between two points for them to be considered part of the same cluster. To determine the appropriate value for *eps*, we plotted the data points to gain a visual understanding of the distribution and density of the data. Based on our analysis, we selected a value of 2 for the *eps* parameter.
- *minPts*: The *minPts* value represents the minimum number of points required to form a cluster. We determined that the appropriate value for *minPts* should be at least 5, based on our domain knowledge. This means that at least 5 similar errors must have occurred for them to be considered a misconception and to form a cluster.

IV. RESULTS AND DISCUSSION

In this study, we do not do any post-processing, so the results we show are based on the raw output.

A. Clustering with Expression Trees

For the first test, we used the same algorithm and parameters as used by [9] (categorical clustering using k-modes), except for the cluster number.

The difference in our approach lies in the input format; here, equations are represented as expression trees using postfix notation, as explained in Section III-B. We used 200 clusters and observed that the output was much more homogeneous than the results from [9]. Figure 1 shows a part of the results from this implementation, displaying a few clusters representing different types of misconceptions. It is clear that the data within each cluster have the same structure, which is a satisfactory result as the goal is to identify common mistakes.

An important finding during data analysis was the presence of a large cluster (cluster 0) containing approximately 10% of the total dataset with 110 entries out of 1062. This could be attributed to the limited number of generated clusters, which resulted in consolidating the residual data into a single cluster. In comparison, the other clusters' average size was approximately six entries.

Also, it is essential to emphasize that the limitations cited by [9] are mostly solved. Based on this result, we can observe that the use of the expression tree, besides solving some issues found in [9], does not affect the algorithm's performance.

B. Clustering and Natural Language Processing

Our second approach was to use the NLTK Tokenizer to transform our textual results (the tree representation) into a numerical form based on tokens, to be able to use the DBSCAN algorithm that works only with numerical entries. We found 178 clusters by using the NLP layer, which made the clusters more realistic and similar. This makes it easier to correlate the results with a real-life learning scenario. Figure 2 shows part of these results.

The use of expression trees and the NLTK Tokenizer helped address the previously described limitations related to the equation's structure. Consequently, analyzing the output clusters made it easier to identify the patterns. Furthermore, the noisy outputs were limited to unique patterns in the dataset. With these changes, the results resemble what would happen in real-life, and it is easier to identify misconceptions.

With this method, we were able to identify more complex misconceptions compared to the k-modes method, which grouped most of the complex equations into one single cluster.

V. CONCLUSION AND FUTURE WORKS

Misconceptions are common in mathematics education, where students may develop an incorrect understanding of mathematical concepts and principles for various reasons. This study proposes two new approaches to identify misconceptions in mathematics, improving the method proposed by [9]. The presented techniques aim to provide a more accurate and effective way of identifying these misconceptions to help students overcome them.

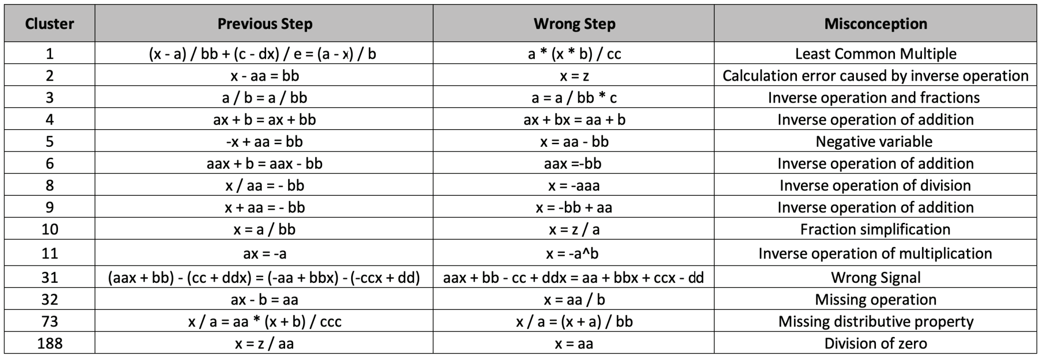
In this study, we presented and compared the following methods:

Clustering an equation represented by expression trees using K-modes: In this approach proposed by [9], each equation is represented as an expression tree. The k-modes algorithm is then used to cluster the expression trees based on their structure.

Clustering a tokenized equation represented by an expression tree using DBSCAN: This approach involves, in addition to creating the expression trees, tokenizing equations into their constituent parts, such as operators, variables, and coefficients, and then using the DBSCAN algorithm to cluster the tokens.

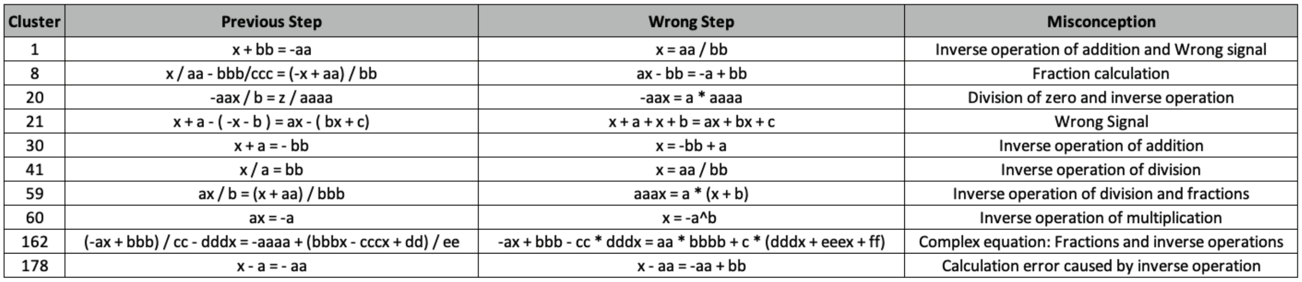
After conducting a manual visual inspection, we could identify that the second approach, which involves clustering tokenized expression trees with DBSCAN, was more successful in grouping similar misconceptions. This resulted in more cohesive and meaningful clusters. By utilizing tokenization and DBSCAN, we were able to more accurately identify patterns and relationships within the data, which ultimately led to a more precise identification of misconceptions.

These findings may offer a valuable contribution to the field of mathematics education, as they provide a more accurate and in-depth understanding of student misconceptions. However, the data still requires further analysis and interpretation by experts in the field to be useful. Future research directions include the use of more sophisticated algorithms and data



Cluster	Previous Step	Wrong Step	Misconception
1	(x - a) / bb + (c - dx) / e = (a - x) / b	a * (x * b) / cc	Least Common Multiple
2	x - aa = bb	x = z	Calculation error caused by inverse operation
3	a / b = a / bb	a = a / bb * c	Inverse operation and fractions
4	ax + b = ax + bb	ax + bx = aa + b	Inverse operation of addition
5	-x + aa = bb	x = aa - bb	Negative variable
6	aax + b = aax - bb	aax =-bb	Inverse operation of addition
8	x / aa = - bb	x = -aaa	Inverse operation of division
9	x + aa = - bb	x = -bb + aa	Inverse operation of addition
10	x = a / bb	x = z / a	Fraction simplification
11	ax = -a	x = -a^b	Inverse operation of multiplication
31	(aax + bb) - (cc + ddx) = (-aa + bbx) - (-ccx + dd)	aax + bb - cc + ddx = aa + bbx + ccx - dd	Wrong Signal
32	ax - b = aa	x = aa / b	Missing operation
73	x / a = aa * (x + b) / ccc	x / a = (x + a) / bb	Missing distributive property
188	x = z / aa	x = aa	Division of zero

Fig. 1. Results using expression trees



Cluster	Previous Step	Wrong Step	Misconception
1	x + bb = -aa	x = aa / bb	Inverse operation of addition and Wrong signal
8	x / aa - bbb/ccc = (-x + aa) / bb	ax - bb = -a + bb	Fraction calculation
20	-aax / b = z / aaaa	-aax = a * aaaa	Division of zero and inverse operation
21	x + a - (-x - b) = ax - (bx + c)	x + a + x + b = ax + bx + c	Wrong Signal
30	x + a = - bb	x = -bb + a	Inverse operation of addition
41	x / a = bb	x = aa / bb	Inverse operation of division
59	ax / b = (x + aa) / bbb	aaax = a * (x + b)	Inverse operation of division and fractions
60	ax = -a	x = -a^b	Inverse operation of multiplication
162	(-ax + bbb) / cc - dddx = -aaaa + (bbbx - cccx + dd) / ee	-ax + bbb - cc * dddx = aa * bbbb + c * (dddx + eeex + ff)	Complex equation: Fractions and inverse operations
178	x - a = - aa	x - aa = -aa + bb	Calculation error caused by inverse operation

Fig. 2. Results using DBSCAN and expression trees

visualization tools to better understand and address these misconceptions.

Furthermore, our objective is to automate this process entirely, and we plan to utilize post-processing in future research to identify common mistakes and present them in a user-friendly manner. We could explore the following approaches to achieve this goal: 1) interpret the clusters and the related misconceptions, and offer students helpful tips and supplementary resources, or 2) convert the equations to infix notation and demonstrate the proper method of solving them.

To further improve our approach, we plan to test other clustering methods that can better capture the inherent structures and patterns in the data. One possible approach is to compare the expression trees using tree comparison methods, such as the Tree Edit Distance [13] or the Maximum Common Subtree [14]. These methods can quantify the similarities and differences between trees based on their structural properties and can be used to group trees with similar structures together. Overall, we believe that these efforts will lead to a more accurate and effective method for identifying math misconceptions in real student data.

References

[1] H.-J. Schmidt, "Students' misconceptions—looking for a pattern," *Science education*, vol. 81, no. 2, pp. 123–135, 1997.

[2] Y. Ay, "A review of research on the misconceptions in mathematics education," *Education Research Highlights in Mathematics, Science and Technology*, vol. 2017, pp. 21–31, 2017.

[3] M. Elmadani, M. Mathews, and A. Mitrovic, "Data-driven misconception discovery in constraint-based intelligent tutoring systems," in *Int. Conf. on Computers in Education*, Singapore, 2012.

[4] J. Hattie and H. Timperley, "The power of feedback," *Review of Educational Research*, vol. 77, no. 1, pp. 81–112, 2007.

[5] D. Kluger and A. Z. N. Klein, "The power of feedback revisited: A meta-analysis of educational feedback research," *Frontiers in Psychology*, vol. 11, p. 3087, 2020.

[6] P. A. Jaques, H. Seffrin, G. Rubi, F. de Morais, C. Ghilardi, I. I. Bittencourt, and S. Isotani, "Rule-based expert systems to support step-by-step guidance in algebraic problem solving," *Expert Systems with Applications*, vol. 40, no. 14, pp. 5456–5465, 2013.

[7] M. Q. Feldman, J. Y. Cho, M. Ong, S. Gulwani, Z. Popović, and E. Andersen, "Automatic diagnosis of students' misconceptions in k-8 mathematics," in *ACM CHI*. Montréal, CA: ACM, 2018, p. 264.

[8] J. Andersson and H. Johansson, "Using clustering in a cognitive tutor to identify mathematical misconceptions," *LU-CS-EX 2015-45*, 2015.

[9] J. C. Gomes and P. A. Jaques, "A data-driven approach for the identification of misconceptions in step-based tutoring systems," in *Brazilian Symp. om Computers in Education*. SBC, 2020, pp. 1122–1131.

[10] G. Lample and F. Charton, "Deep learning for symbolic mathematics," *arXiv preprint arXiv:1912.01412*, 2019.

[11] S. Bird, "Nltk: the natural language toolkit," in *Proceedings of the COLING/ACL 2006 Interactive Presentation Sessions*, 2006, pp. 69–72.

[12] P. A. Jaques, M. Lehmann, and K. S. F. Jaques, "Avaliando a efetividade de um agente pedagógico animado emocional," in *Brazilian Symp. on Computers in Education*, vol. 1, no. 1, 2008, pp. 145–154.

[13] K. Zhang and D. Shasha, "Simple fast algorithms for the editing distance between trees and related problems," *SIAM journal on computing*, vol. 18, no. 6, pp. 1245–1262, 1989.

[14] A. Kundu, S. K. Pal, and D. D. Majumder, "Comparison of tree structures: A survey and some experimental results," *Int. J. of Computer Science and Network Security*, vol. 10, no. 7, pp. 154–164, 2010.