mathematics

MDPI

Article

Discovery Model Based on Analogies for Teaching Computer Programming

Javier Alejandro Jiménez Toledo [1,*], César A. Collazos [2] and Manuel Ortega [3]

[1] Faculty of Engineering, Systems Engineering, CESMAG University, Pasto 520001, Colombia
[2] System Department, Faculty of Electronic Engineering and Telecommunications, University of Cauca, Popayán 190001, Colombia; ccollazo@unicauca.edu.co
[3] Department of Technologies and Information Systems, Higher School of Informatics, Castilla-La Mancha University, 13001 Ciudad Real, Spain; manuel.ortega@uclm.es
* Correspondence: jajimenez@unicesmag.edu.co; Tel.: +57-310-495-4795

Abstract: Teaching the fundamentals of computer programming in a first course (CS1) is a complex activity for the professor and is also a challenge for them. Nowadays, there are several teaching strategies for dealing with a CS1 at the university, one of which is the use of analogies to support the abstraction process that a student needs to carry for the appropriation of fundamental concepts. This article presents the results of applying a discovery model that allowed for the extraction of patterns, linguistic analysis, textual analytics, and linked data when using analogies for teaching the fundamental concepts of programming by professors in a CS1 in university programs that train software developers. For that reason, a discovery model based on machine learning and text mining was proposed using natural language processing techniques for semantic vector space modeling, distributional semantics, and the generation of synthetic data. The discovery process was carried out using nine supervised learning methods, three unsupervised learning methods, and one semi-supervised learning method involving linguistic analysis techniques, text analytics, and linked data. The main findings showed that professors include keywords, which are part of the technical computer terminology, in the form of verbs in the statement of the analogy and combine them in quantitative contexts with neutral or positive phrases, where numerical examples, cooking recipes, and games were the most used categories. Finally, a structure is proposed for the construction of analogies to teach programming concepts and this was validated by the professors and students.

Keywords: machine learning; modeling; programming; text analysis

Citation: Jiménez Toledo, J.A.; Collazos, C.A.; Ortega, M. Discovery Model Based on Analogies for Teaching Computer Programming. *Mathematics* **2021**, *9*, 1354. https://doi.org/10.3390/math9121354

Academic Editor: Radi Romansky

Received: 28 March 2021
Accepted: 9 June 2021
Published: 11 June 2021

1. Introduction

Learning to program computers is a difficult process for novice students and also for professors [1] as it not only means acquiring new knowledge, but also fundamentally applying this knowledge to solve problems [2]. Therefore, it becomes a complex task because it requires students to master higher-order cognitive skills [3].

On the other hand, analogies are didactic tools used very frequently in the educational sector, and from observed experience, it can be determined that some professors include them in their curricula in a planned way and others adopt them only as part of their explanations in the classroom.

The scientific literature reports few experiences in the use of analogies for teaching-learning in a CS1, one of which is Collece 2.0, a programming learning environment that has been extended with mixed reality techniques and uses the analogy of roads and traffic signs for the study of recursion concepts [4–6]. In turn, Sukamto and Megasari [7] developed a model that converts source code into analogy images using state machines for teaching programming.

Likewise, there are models based on analogies for teaching processes in fields other than a CS1, which is how Strunz and Louie [8] proposed a model of analogy between energy

storage and data processing in a computer for teaching electrical engineering concepts. For his part, Plappally [9] designed a model of analogies using concept maps for the first year of mechanical engineering courses. Additionally, Stockdill et al. [10] implemented a correspondence-based analogy model to choose representations of mathematical problems, among others.

Taking into consideration the ease of incorporating analogies in a teaching–learning process and the difficulties reported in the scientific literature in a CS1, this study proposed a model of knowledge discovery based on machine learning and text mining, which took as input the didactic teaching strategy with analogies and obtained a set of patterns for possible scenarios that can be used with a greater degree of assertiveness when presenting an analogy on a CS1.

In the collection of information, a semi-structured questionnaire was used to characterize the formal and non-formal analogical processes developed in a CS1 class with university professors. This questionnaire collected information from the professor, university, course, and the analogies used in teaching for four fundamental concepts: entries, exits, conditionals, and cycles. The application of this survey reported a total of 570 examples of analogies in 15 universities of five countries (Mexico, Ecuador, Argentina, Spain, and Colombia) with a total of 33 expert professors in CS1 in university computer programs that train professionals in software development.

The proposed knowledge model has three stages: initial processing, discovery, and results. In the initial processing stage, three activities were developed: data collection, extraction, and pre-processing. Two major activities were carried out in the discovery stage: transformation and mining. Finally, in the results stage, the findings found in the mining process were complemented, for this, techniques of the Natural Language Toolkit (NLTK) were implemented through three tasks: linguistic analysis, textual analytics, and experimental analysis.

The rest of this paper is organized as follows. In Section 2, the conceptual foundation of this study is presented. Section 3 describes the proposed discovery model. Section 4 shows the results obtained when applying the discovery model. In Section 5, the results are discussed and a structural model is presented to build analogies to teach programming concepts, which was validated through an experiment with professors and students. Finally, Section 6 presents the conclusions and future work.

2. Literature Review

The teaching–learning processes of computer programming are a complex task, and the results of teaching programming show that professors do not generate a reflection process in problem solving [11], despite the fact that many students do not have previous training experience in this field [12]. This is even more so when facing a CS1 as it requires cognitive abstraction skills, logical-mathematical skills, and the ability to solve problems algorithmically [13].

On the other hand, the learning of programming through analogous representations is a process where work has been underway for some years and whose purpose is to reduce the level of abstraction that programming requires to facilitate its understanding [4,14].

This is how an analogy is a connection of two situations with common relationship patterns between them and they are bidirectional [15]. Analogy can be explanatory when it poses new concepts and principles in familiar terms and creative when it stimulates the solution and the identification of a new problem and the generalization of knowledge [16].

The word analogy was initially a mathematical concept that meant proportion [17] and it was later considered as not corresponding to an identity of two relationships but rather ensures a similarity of correlations [18]. That is to say, it does not imply symmetric equality, but a relationship used with the purpose of clarifying, structuring, and evaluating the unknown from what is known [19].

Through analogies, one can develop the creativity, imagination, skills, and attitudes necessary for the critical use of scientific models and to shape one's own reality [20,21].

Furthermore, analogies can stimulate the professor-researcher to take into consideration the students' prior knowledge [22].

Analogies are a relevant research topic in the field of teaching [23]. Currently, there is an analog didactic model (ADM), which helps students find concepts that already exist in their cognitive structures on which a new learning is built [24].

Alternatively, artificial intelligence (AI) is a simulation of the human intelligence process in machines [25,26], where one of its most important fields is machine learning, which together with natural language processing (NLP) are contributing significantly to the development of science through research [27].

In addition, Knowledge Discovery in Databases (KDD) is a process that facilitates findings and analysis with the purpose of extracting unusual patterns in the form of rules or functions [28–30] through techniques that include data mining (DM) and text mining (TM) [31] as fundamental elements of machine learning.

Despite the beginning of machine learning dating back several decades, it is currently considered as an emerging field for developing research processes [32], demonstrating unexpected results in complex situations that resemble processes developed by human experts or superior to them [33].

Machine learning generates learning with low computational complexity [34] by which it is possible to extract behavior patterns from a dataset and build predictive models [35] using two phases in data processing: training and testing [33]. Furthermore, the training data requires standardization processes to ensure its efficiency [36]. An important feature of machine learning is the continuous upgrade, which can lead to changes in training data [37].

Additionally, machine learning takes into consideration three types of learning: supervised, unsupervised, and semi-supervised [38,39]. In supervised learning, classification, regression, and prognosis tasks are identified with techniques such as trees and decision rules, neural networks, support vector machines, and Bayesian classifiers, among others. Unsupervised learning features grouping tasks, association, and reduction with techniques such as clustering and dimensionality reduction techniques, while techniques such as transductive support vector machines and hope maximization are found in semi-supervised learning [40–42].

Another important area of machine learning is deep learning [43,44], natural language processing (NLP) [25], and text mining, which are computer and statistical techniques developed in the early nineties [45] used to discover new non-explicit information in source documents [46,47] and was aimed at discovering patterns through the right combination of artificial intelligence, machine learning, statistics, and data mining [48]. Text mining is currently used for various operations such as in image analysis [49], in biomedical fields (hospitals and clinics) [50,51], in the analysis of profiles of university students [52], in geosciences (geology, geophysics, geochemistry and remote sensing) [53], and in the automotive sector [54], among others.

There are several moments in the methodological processes formulated in text mining. Verma, Ranjan, and Mishra proposed two stages: classification and extraction [55]. In contrast, Sukanya and Biruntha established five different steps: purpose of study, information retrieval, processing, extraction, and results [56]. Similarly, Miner et al. proposed four tasks: collecting, organizing, analyzing, and assimilating [57]. Uysal and Gunal introduced four stages: pre-processing, extraction, selection, and classification [58]. In addition, text mining incorporates data mining tasks that are also used in machine learning within its learning types. These tasks are association, classification, and segmentation [27,28,41,59].

3. Materials and Methods

In this study, a knowledge discovery model based on machine learning and text mining is proposed. It takes the didactic teaching strategy as an input together with analogies for processes of abstraction of the fundamental concepts in a CS1 developed by

professors in their classes. This model (see Figure 1) consists of three stages: (1) Initial processing; (2) discovery, and (3) results.

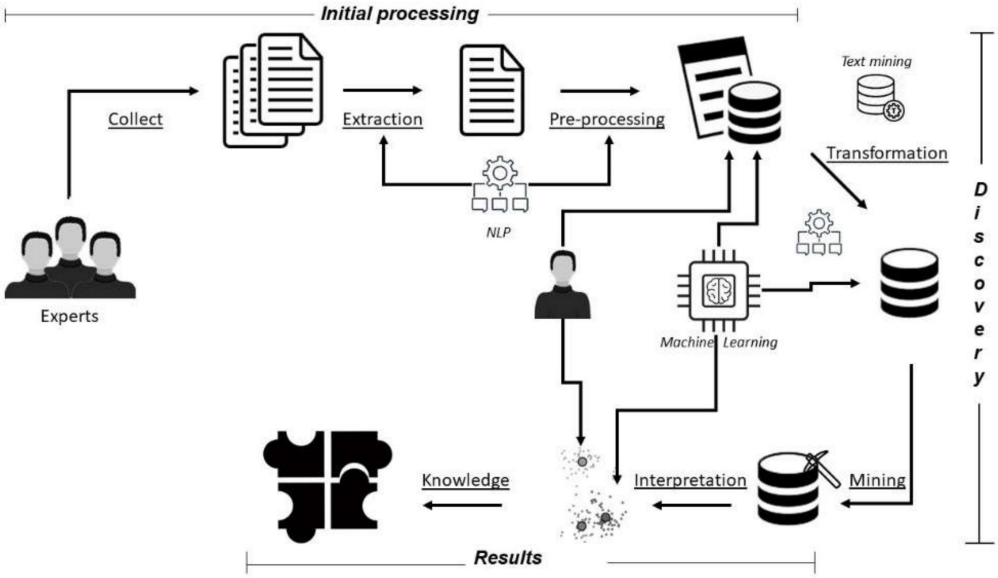

Figure 1. Knowledge discovery model.

3.1. Initial Processing Stage

The initial processing stage consists of three activities: Data collection, extraction, and pre-processing. For the collection, the survey was used as a technique and a semi-structured questionnaire was constructed in which the objective was to characterize the formal and non-formal analog processes developed in class for a CS1 with university professors. This survey was divided into two parts. The first part was related to the general characterization data of the professor, the university, and the course. In the second part, the experience of incorporating analogies in four fundamental concepts was investigated: inputs, outputs, conditionals, and cycles.

The extraction activity allowed us to carry out a characterization of professors and courses, in addition, it classified the examples of analogies in each of the following thematic units: analogies for inputs, outputs, simple conditionals, compound conditionals, nested conditionals, selective structures, cycle for, cycle while, and cycle do while. In this activity, there were a total of 570 examples of analogies used by professors in a CS1.

In the pre-processing activity, a dataframe was built under a data matrix structure and a cleaning process was carried out using NLTK [60] that consisted of: eliminating white spaces at the beginning, at the end, and between words, eliminating special characters, converting all words to lowercase, removing spelling accents, identifying words with plural and singular, and correct misspelled words.

3.2. Discovery Stage

Two major activities were carried out in the discovery stage: transformation and mining.

3.2.1. Transformation Activity

During the transformation activity, the text mining technique called the semantic vector space model [61] was used through the distributional semantics approach Gensim [62] that analyzes the words individually using: keyword extraction, summary extraction, tokens, and tagging part of speech (PoS).

With the previous activities, the following grammatical elements were created: main verbs and noun. Moreover, with the participation of three experts, a classification technique was applied under the intent detection model [63], which allowed us to extract the following semantic fields: action, context, and category.

Finally, during the transformation, it was necessary to resort to the synthetic data generation method [64] to balance the dataset using Python's NLTK to generate synonyms for each contemplated variable, which allowed for balancing the number of tuples of the input and output concepts versus conditionals and cycles.

3.2.2. Mining Activity

Before starting the mining activity, it was necessary to split the dataset into two: training (to build the model) and testing (to evaluate its behavior). Therefore, stratified sampling was used and the variable "Concept" (input, output, conditional, and cycle) was taken as a parameter. With these considerations, a setting of 80% for the training dataset and 20% for testing was done using cross validation to obtain better results [65].

In turn, the purpose of the mining activity was to identify patterns that allow the professor to use appropriate analogies to teach fundamental concepts of computer programming in CS1. In that way, and to obtain better results in the detection of patterns, three types of learning were applied: supervised, unsupervised, and semi-supervised learning with their corresponding tasks [66].

In the mining activity, the balanced dataset was used and began with the supervised type of learning through the classification task [67]; as it is one of the most widely used, it has great importance in this study due to its purpose of classifying data into categories that allow for the discovery of patterns in a CS1.

In contrast, the type of unsupervised learning was initiated through the association task [68] with the purpose of discovering frequent actions that were present in the bank of analogies used by professors in CS1. There were two methods used for this task: Apriori (to search for groups of frequent items) and Predictive Apriori (to extract the best rules with support and trust parameters).

Finishing with the mining activity and in order to obtain more information from the set of examples collected as analogies used in a CS1, the type of semi-supervised learning was applied using the expectation maximization technique (EM) [69] that obtains groups using a probabilistic approach.

3.3. Results Stage

Once the mining activity was completed and to strengthen the pattern detection process, the findings were complemented with three additional techniques: linguistic analysis [70], textual analysis [71], and experimental analysis [72] for each of the four main concepts established in this study.

For linguistic analysis, the techniques of flexive lemmatization and morphosyntactic labelling [73] were used. For the textual analysis, extraction techniques were used which involved keywords, multi-word, entities, and sentiment analyzer [74]. Finally, in the experimental analysis, the technique of extracting data linked by triplets was used [75].

Consequently, for the interpretation of results, the findings of the three techniques described above and the results of the mining stage were used, supported under the semantic vector space model with Tf-idf [61] with the Python OpenCv library. In addition, scripts were used to generate frequency graphs, point diagrams, word clouds, hierarchical grouping with elimination of dispersed terms and visualization of dendrograms by levels.

Likewise, with WEKA, the different cluster display models, trees, and margin curves were implemented. With these tools, it was possible to complement the process of the extraction of patterns from the bank of analogies used in teaching a CS1 by the 33 professors involved in this study.

Finally, an experiment was conducted in two moments, first by professors of a CS1, and then with their students.

4. Results

This section presents the results obtained in each stage of the knowledge model proposed in this study.

4.1. Initial Processing Stage

There were 15 state and private universities in five countries (Mexico, Ecuador, Argentina, Spain, and Colombia) that participated in this process, with a total of 33 professors who are experts in CS1 in university computer programs that train professionals in software development.

After the completion of the data collection, the extraction activities began. They, revealed that out of the 33 experts, 20 were from state universities and 13 from private ones. In addition, 21% of them had a doctoral degree, 70% had master's degree and 9% were specialists. Furthermore, in the characterization of courses, it was seen that the courses with the highest enrolment were Introduction to Programming, Algorithms and Programming, Programming I, and Fundamentals of Programming, which corresponded to 82% of the name of the courses. Additionally, the names of the professional programs that offer these courses corresponded to Systems Engineering, Computing, and Computer Engineering, which represent 95%. Finally, all the courses consulted were taught in the first semester of each academic program.

The pre-processing activity ends with the construction of the dataframe that had the following variables: Name of the university, country where the university is located, type of university, last study done by the professor, name of the course, semester in which it is taught, program or faculty, description of the analogy, concept that was intended to be taught, and if it was used directly in classes. Table 1 shows some analogies described by the professors at this stage.

Table 1. Some analogies used in a CS1.

Analogy	Concept
To Add an ingredient to prepare food	Input
To Enter password in ATM	
To Print sales invoice	Output
To Calculate total to pay	
To Choose menu of the day	Conditional
To Evaluate based on age, whether you are younger or older	
To Generate the first ten numbers	Cycle
To Add products in a cash register	

4.2. Discovery Stage

4.2.1. Transformation

With the results of the pre-processing and transformation activities, a relational database model was built (implemented in MySQL) with the following entities: Countries, Universities, Types Universities, Departments, Courses, Professors, Analogies, Concepts, and Descriptors.

The transformation activity incorporated the creation of datasets through dynamic views that allowed for the inclusion of the fields required for the construction of training algorithms necessary for the mining phase.

Before carrying out the mining activity, a pre-analysis was made using NLTK with the first data contained in the initial dataset, where the following information was obtained: Number of records for each concept: input (57), output (43), simple conditional (76), compound conditional (72), nested conditional (64), multiple selective (64), cycle for (66), cycle while (69), and cycle do while (59).

Additionally, this dataset contained the following peculiarities: out of the 570 cases of analogies used by professors, 523 were used in class with students and 47 were proposed because the professor had not used them for a certain concept and they suggested that they

could be used. It was also observed that the same analogy was used to explain more than one concept.

Likewise, the most widely used contexts in the analogies for "Inputs" were Food, Shopping and Recipes. For "Outputs", the contexts used were Product, Invoices, and Purchases while for "Simple Conditional", the contexts used included Climate, Purchase, and Clothing. For "Compound Conditional", Transportation, Age, and Supermarket were used. Purchases, Salary, and Dress were the contexts used for "Nested Conditional". For "Selective Structure", Calculator, Food, and Cashier were used whereas Multiples, Students, and Clock were the contexts used for "Cycle For". For "Cycle While", Students, Play, and Multiples were the contexts used while Game, Students, and Shopping were used for "Cycle Do While".

A total of 116 different verbs were also found and the most frequent was "Evaluate". There were 309 nouns where the most widely used was "Age". Additionally, there were 402 actions and the most frequent was "Evaluate age". There were 272 contexts and the one with the highest concurrence was "Game", and finally, there were 11 categories where the most repeated was "Activity".

These findings applied to a CS1 are related to the teaching–learning processes of programming in studies carried out in children as presented by Pérez et al. [76,77], Sukamto and Megasari [7], or through metaphors by Chibaya [78].

4.2.2. Mining

Supervised Learning

The mining activity started with the type of supervised learning through the classification task and with the purpose of obtaining an appropriate model to the training dataset. In this task, the "general concept" attribute was chosen as the class variable that categorizes the analogies for inputs, outputs, conditionals, and cycles. In addition, there were five techniques used: decision trees, classification rules, vector support machines, envelope classifiers, and meta classifiers. The implementation of these methods was done with Weka.

There were two methods used for the decision tree classification task: J48 (which in turn contains the decision tree algorithm C4.5) and logistic model trees (LMT).

Due to the size of the dataset, the trees generated were complex to analyze due to the number of leaves and the size of the tree, therefore, it was necessary to use three pre-pruning techniques, taking into consideration to not affect the threshold of the percentage of instances classified correctly. The first pruning technique consisted of manipulating the confidence factor in each node (CF); the second by means of the minimum number of instances per leaf (IH); and the third by the elimination of attributes (EA).

Thus, a set of training models was planned by applying values from 10% to 40% to the CF; for IH, there were values from 2 to 40 records and EA up to three negations of the main attribute, for a total of 48 analyzed trees. In addition, a post-pruning process was also necessary. This consisted of discarding the generated rules that were under the established threshold of trust and support of the data.

As a result of the pre-pruning process, one of the generated decision trees that considered attributes of action, verb, noun, and context, correctly classified 93.3% of the instances with a minimum confidence of 83.9%. This tree took the verb attribute as the root node and the general rules with the highest value are shown in Table 2.

Table 2. Pre-pruning with J48.

Rule	Concept	% Confidence	Records by Rule
Verb = 'to enter'	Input	83.9	31
Verb = 'to get'	Output	88.6	21
Verb = 'to evaluate'	Conditional	97.5	193
Verb = 'to do'	Cycle	91.2	131

The J48 and LMT methods in the pre-pruning process classified the following verbs: to enter, to register, to obtain, to calculate, to evaluate, to do, and to play and the arithmetic and shopping contexts with the highest scores. In the post-pruning process, important variables related to noun, action, and context are evidenced, some results were: Noun (ingredient, row, plate, area, higher, assistance, products, budget, etc.), action (insert, ingredient, get, receive money, assess age, evaluate number, make multiples, play numbers, etc.), and context (recipe, bus, product, factory, students, purchases, game, students, etc.)

For the classification task using rules, four methods were employed. They were the Ripper algorithm (Jrip), machine learning sequential coverage algorithms (Modlem), One Rules based on ID3 (OneR), and Part Decision List based on C4. 5 (Part). In total, 678 rules were obtained.

In turn, the classification by means of vector support machines was performed with the SMO method, which is a sequential algorithm of minimum optimization through which a total of 1756 rules were obtained.

In the classification using envelopes, the input mapped classifier method was used taking into consideration the training and test dataset. This algorithm classified 21 rules.

Additionally, using the IterativeClassifierOptimizer Meta classifier, 10 iterations were performed with cross validation, obtaining 40 rules.

Finally, the results for supervised learning in the classification task were obtained considering the precision indicators, area under the curve, classification, and the positive rate for each of the methods used. The indicators in the classification task for the inputs, outputs, conditionals, and cycles concepts are presented in Table 3.

Table 3. Correctly Classified Instances.

Method	Input %	Output %	Cond. %	Cycle %
J48	98.4	98.4	98.4	98.4
LMT	94.4	94.4	94.4	94.4
Jrip	81.1	81.1	81.1	81.1
Modlem	99.1	98.9	99.1	99.1
OneR	98.6	98.6	98.6	98.6
Part	80.5	86.1	80.5	86.1
SMO	86.9	86.9	86.9	86.9
InputMappedClassifier	97.7	97.7	97.7	97.7
IterativeClassifierOptimizer	85.8	85.8	85.8	85.8

The most important findings of supervised learning for each of the concepts established in this study are presented below.

In the bank of analogies proposed for the input concept, the most important classification rules obtained in the nine methods were:

If [(verb in {enter} & noun in {ingredient, food} context in {recipe})] or [(verb in {enter, place} & context in {recipe, salary, arithmetic, student, price})] or [(verb in {register, type, read} & context in {salary, arithmetic, student, price})] then Input.

In the analogies proposed for the output concept, the most important classification rules obtained in the nine methods were:

If [(verb in {get} & noun in {ingredient, food} context in {recipe})] or [(verb in {get, calculate} & noun in {total, area, value} & actions in {get result})] or [(verb in {print, compute} & context in {arithmetic})] then Output.

Likewise, the most important classification rules for the concept of conditional were:

If [(verb in {choose} & noun in {food, time} & context in {food, restaurant})] or [(verb in {evaluate} & context in {mathematical, student} & noun in {age, number, grade, price})] or [(verb in {choose, determine, evaluate} & context in {arithmetic})] or [(action in {evaluate # noun})] then Conditional.

Furthermore, the most important classification rules for the concept of Cycle were:

If [(verb in {do, count, repeat} & noun in {age, multiple})] or [(verb in {add, average} & actions in {add ages, add wages, average age})] or [(verb in {add, count} & context in {arithmetic, student})] or [(noun in {row, money, bill, energy, shift} & context in {shopping, cashier, playing, television})] then Cycle.

Unsupervised Learning

This was started by the association task through two methods: Apriori and Predictive Apriori.

During the generation of rules with Apriori, a minimum support of 1% was used with a confidence of at least 70% and in total, 328 rules were obtained of which the main ones are shown in Table 4.

Table 4. Association with Apriori.

Concept	Rule
Input	(verb in {enter} & (noun in {ingredient}))
Output	(verb in {get, generate, print})
Conditional	(verb in {choose, evaluate, determine} & (noun in {operation, number, note, options, food} or (context in {purchase, food, student})))
Cycle	(verb in {make, play, count, assemble, add} & (noun in {multiples, vehicle} or (context in {arithmetic}))

In the search for rules with Predictive Apriori, the same parameters of support and confidence were configured as Apriori. Thus, we obtained 361 association rules.

This was then complemented with the grouping or segmentation task, which allows intra-group differences to be minimized and extra-group differences to be maximized.

Before applying this technique, a scaling, weighing, and selection process was performed with the data in order to obtain a more reliable segmentation level. For the grouping task, the K-means technique was used to search for characteristics iteratively.

This technique was configured with a value of 120 as seed in the generation of random numbers and four for the number of groups. For K-means, the grouping for inputs, outputs, conditionals and cycles according to the data classified in the dataset had a success of 89%, 91%, 99%, and 85%, respectively.

The most important findings of unsupervised learning are presented below.

The results for unsupervised learning in the association task for the input concept obtained the following rules:

If (verb in {enter} & noun in {ingredient, key}) then Input.

If [(verb in {enter} & context in {purchase})] or [(verb in {read, enter, type, insert} & noun in {price, side, key, money, card, code})] or [(verb in {digit, insert} & noun in {arithmetic})] or [(verb in {read, enter} & noun in {student})] or [(verb in {enter, enter} & context in {food})] then Input.

Similarly, the association task for the output concept generated the following rules:

If [(verb in {get, generate, print})] or [(verb in {get, generate, receive, finish} & (noun in {plate, product, information} or context in {invoice, information}))] or [(verb in {print} & context in {invoice, document, arithmetic, calculation})] then Output.

Likewise, the most important conditional rules for the association task were:

If [(verb in {choose, evaluate, determine} & noun in {operation, number, note, purchase, options, food, weather, age} or context in {purchase, food, calculator, student})] or [(verb in {evaluate, choose, determine, validate} & noun in {number, note, purchase, food, color, age, climate, price} or context in {age, climate, note, purchase, salary})] or [(verb in {choose} & noun in {menu, food} & context in {restaurant})] then Conditional.

The most important cycle rules for the association task were:

If [(verb in {make, play, count, assemble, add} & noun in {multiples, vehicle} or (context in {arithmetic}))] or [((verb in {play, count, do, add, run}) or (noun in {multiples,

student, product}) or (context in {arithmetic, student, budget}))] or [(verb in {play} & noun in {numerical care, professions, parquet, ballot, Tingo_Tango, hideout, war fest, cards, Cucunova, Hanoi, guess_number})] then Cycle.

Finally, for unsupervised learning in the grouping or segmentation task and in order to determine similar characteristics of the input concept, a new dataset was generated with only the corresponding tuples and to which the input class attribute was discarded, therefore, the K-means technique was configured to partition them into three groups. Correspondingly, three more datasets were made with the same criteria for the concepts of output, conditional, and cycle. The most important results can be observed in Table 5.

Table 5. Grouping by the K-means concept.

Concept	Variable	Cluster 0	Cluster 1	Cluster 2
Input	Verb	To enter	To read	To register
	Context	Recipe	Arithmetic	Purchases
Output	Verb	To get	To calculate	To get
	Context	Arithmetic	Purchases	Recipe
Conditional	Verb	To choose	To evaluate	To decide
	Context	Various	Various	Various
Cycle	Verb	To do	To tell	To do
	Context	Calculations	Various	Activities

Semi Supervised Learning

EM was configured with the same seed value as K-means and although the cluster number was not configured in EM, it generated four of these. The most important results for the semi-supervised learning in the grouping or segmentation task with the EM technique and with the same methodology carried out with K-means with four. The dataset is shown in Table 6.

Table 6. Grouping by the EM concept.

Concept	Variable	Cluster 0	Cluster 1	Cluster 2
Input	Verb	Various	To enter	To enter
	Context	Activity	Recipe	Purchases
Output	Verb	To get	To generate	Various
	Context	Arithmetic	Action	Vehicle
Conditional	Verb	To evaluate	To evaluate	To evaluate
	Context	Numeric	Activity	Calculator
Cycle	Verb	To do	To play	To tell
	Context	Numeric	Games	Numeric

4.3. Results Stage

Once the mining activity was concluded and using NLTK, three tasks were applied: linguistic analysis, textual analytics, and experimental analysis for each of the four main concepts established in this study.

4.3.1. Linguistic Analysis

This activity was developed with NLTK and its purpose was to extract information from the corpus of the analogies analyzed here to complement the patterns obtained in the discovery phase.

As a result of the linguistic analysis in the analogies described by the professors to face the abstraction of the input concept, there was a lexical variety of 81% where the use of verbs was 12.2% and nouns 43.5%. The main results for this analysis were:

Verbs: To enter, to register, to read, to place, to fill, to insert, to type, and to start. Additionally, the most frequent nouns were: ingredient, side, row, food, object, key, data, money, article, card, code and day. At the same time, the most recurring actions were: adding an ingredient, entering a password, reading text, entering a bus, turning on a circuit, filling a locker, queuing, filling fuel, and entering data. Regarding the context, the most used were: recipe, bus, arithmetic, student, data, restaurant, recipe, telephone, salary, and price. Figure 2 shows the morphosyntactic labelling process for the "key in cash machine" analogy.

(T) Head position	(T) Head	(T) Head category	(T) Dependent position	(T) Dependent	(T) Dependent category	(T) Syntactic relation
0	Key	noun	2	cash machine	noun	Nominal complement

Figure 2. Input morphosyntactic labelling.

Similarly, linguistic analysis in analogies to teach the concept of output had a lexical variety of 71.2% where the use of verbs was 9.4% and nouns 25.9%. The main results for this analysis were:

The most used verbs were: to obtain, to calculate, to generate, to print and to invoice. The list of most common nouns was: total, area, shake, plate, information, power, shakes, clothing, result, bill, wages, sound, and value. On the other hand, the most mentioned actions were: obtaining a prescription, receiving money, getting off the bus, obtaining a solution, printing an invoice, obtaining an arithmetic result, and obtaining a prepared plate. The most frequent contexts reported were: product, factory, document, oven, arithmetic, printing and invoice.

In the results of the linguistic analysis in the analogies used for the study of conditionals, there was a lexical variety of 61.6% where the use of verbs was 7.8% and 16.9% nouns. The main results for this analysis were:

The most used verbs were: to evaluate, to choose, to determine, to validate, to take, to compare, and to calculate. The nouns were: major, minor, assistance, route, operation, transport, road, qualification, garment, purchase, options, health, color, signal, alternatives and subject, whereas the actions were: evaluate age, evaluate number, evaluate grade, evaluate purchase, choose transportation, choose route, determine age, determine gender, calculate higher, calculate lower, choose case, and evaluate number. The most common contexts included: student, shopping, arithmetic, and transportation.

Likewise, the linguistic results for the study of cycles had a lexical variety of 46.2% where the use of verbs was 6.4% and nouns 12.1%. The main results for this analysis were:

Verbs: to make, to play, to assemble, to count, to add, to perform, to loop, to execute, to count, to average, to determine, to repeat, to rinse, to collect, to put and to convert. The nouns included: product, users, arithmetic, budget, count, activity, game, card, age, multiple, routine, and Fibonacci. The actions were: making multiples, playing numbers, assembling vehicles, counting students, doing routines, doing operations, adding ages, adding wages and playing, whereas the contexts were: game, student, arithmetic, budget, tasks, payroll, cooking, health, reading, sports, counting, and assembling.

4.3.2. Textual Analytics

The main results for the textual analysis for the input concept showed that the most prominent keywords were: ingredient, keyboarding, dispenser and enter. The most frequent multi-words were: cooking recipe, food preparation, geometry figures, side height,

beverage dispenser, bank affairs, quantity of items, and cashier code. Regarding the analysis of entities with the greatest presence in the analogies, there were: ingredient and geometry. The sentiment analyzer indicated that 17.9% of the examples presented were negative phrases, 57.1% were neutral phrases, and 25.0% were positive ones.

Likewise, in the textual analytics for the output concept, the main results of keywords were: plate, arithmetic, operation, and get. The most frequent multi-words were: prepared recipes, geometric figures, receipt, arithmetic operation, calculating salary, informational message, and cashier's money. In the analysis of entities with the highest presence in the analogies were area and concluded. The sentiment analyzer indicated that out of the examples presented, 15.9% were negative phrases, 54.5% were neutral phrases, and 29.6% were positive phrases.

Additionally, the main results of textual analytics for conditionals in terms of keywords were: if, menu, traffic light, cases, validation, and evaluation. The most frequent multi-words were: choice of garment, evaluating option, ideal weight, daily life, choosing destination, type of transport, restaurant menu, and evaluating proposal. In the analysis of entities with highest presence in the analogies were: menu, clothing, and cities. The sentiment analyzer indicated that 17.2% of the examples presented were negative phrases, 32.2% were neutral phrases, and 50.6% were positive phrases.

Moreover, the main results of the textual analysis in the analogies used for cycles provided keywords like: do, summation, factorial, Hanoi towers, iterate, count, play, and while. The most frequent multi-words were: race number, running game, building floor, and multiples. In the analysis of entities with the highest presence in the analogies were: numbers, Fibonacci, iterate, and summations. The sentiment analyzer indicated that 17.5% of the examples presented were negative phrases, 41.8.6% were neutral phrases, and 40.7% were positive phrases.

4.3.3. Linked Data

In the results of data extraction linked by triplets for the input concept, the most important relationships between subject and object were: they will be used for, represent, require, and enter, as shown in Table 7.

Table 7. Triplets.

Subject	Relationship	Object
Ingredients	They will be used to	A meal
Number	It represents	Value on a calculator
The flour	It requires	A baker to make bread
Beverage dispenser	It enters	Money

Similarly, in the extraction of triplets in the output concept, the most important relationships between subject and object were: get a, produce one, and exit.

Likewise, in the conditional triplets, the relations between subject and object with the highest frequency were: if it's there, if it complies, be, and travel in.

Finally, in the extraction of linked data for cycles, the most relevant relationships were: that they are repeated according to, do, add, until, and while.

5. Discussion

It can be seen in the vast majority of the analogies collected in this study that professors wrote them including key verbs that are commonly used in computational terminology and combined them with contexts and actions that are part of people's daily activities. Therefore, the most used categories were: cooking recipes, arithmetic calculations, purchases, sales, school situations, housework, and games, among others.

In addition, the professors used a greater proportion of analogies that could be computationally modeled instead of those that were general and could not be converted into computer programs.

The discussion for each of the three types of learning contemplated in this study is presented in greater detail below.

In supervised learning, professors incorporate verbs such as to enter, to read, to write and to register, to represent an input action, in addition, the examples are presented repetitively in the context of cooking recipes, payroll basic exercises, calculation of arithmetic operations, processing student's grades, transactions and purchases, and actions in electronic devices, among others.

In these rules, it can also be stated that professors include keywords in the use of verbs such as to obtain, to calculate, to generate, to invoice, and to print, which are commonly used in computational terminology to denote an output. In addition, some of the examples found were: obtaining a kitchen recipe, the calculation or printing a payroll, arithmetic, geometric operations, invoices, etc. There were also outputs that represent actions such as obtaining an adequate combination of garments, cars, and clothing, stimuli and the results of processes of electronic devices.

When analyzing the rules found for the analogies used to study the concept of conditional, it is also observed that professors included characteristic verbs of the specific terminology such as to choose, to evaluate, to determine, and to compare. The contexts analyzed were presented when choosing a meal from a menu, evaluating mathematical concepts, student's evaluation aspects, and activities such as doing housework, crossing the street, visiting, traveling, eating, dancing, choice of clothing, etc.

Finally, the rules generated in the analogies used to teach the concept of cycle also included verbs that are closely related to the specific terminology at the time of presenting a computational example for this concept, some of them were do, count, repeat, add, and average. Action such as sums, averages, age counts, wages, multiples, passengers, steps, students, among others, were also recorded.

In unsupervised learning, for the input concept, the association rules obtained were directly related to those extracted with the supervised learning techniques, in which the use of verbs plays an important role in the proposed sentence and with a greater emphasis on nouns in similar contexts.

The association rules for the output concept were more specific than the ones obtained in the classification task and they mainly used the same verbs and contexts. In addition, there were rules that involve actions in general contexts and not purely computational.

The association rules for the conditional concept were more detailed than those generated by classification techniques with a greater participation of nouns and contexts, but sharing the same verbs.

The association rules found in the analogies used for the teaching of cycles contained both the same verbs, nouns, and contexts as the classification rules, but these included more descriptors for verbs and nouns, which allowed us to contemplate a greater number of possibilities when analyzing analogical contexts to approach this concept with new examples.

In semi-supervised learning, the results of the grouping task in Tables 7 and 8 reaffirm the main rules obtained in both the classification and association task previously analyzed. Despite carrying out internal groupings in each concept, subgroups that used the same verbs and contexts were already found.

Table 8. Average grades for the control and experimental groups.

Group	Topic	Pre-Test	Pos-Test
G_1	Input/Output	2.30	4.32
	Conditional	1.70	3.56
	Cycle	2.12	3.87
G_2	Input/Output	2.43	4.03
	Conditional	1.81	3.12
	Cycle	2.40	3.26
G_3	Input/Output	2.60	4.67
	Conditional	2.40	4.26
	Cycle	2.91	4.39
G_4	Input/Output	2.57	4.28
	Conditional	2.13	3.75
	Cycle	2.12	4.00

According to the results found in supervised, unsupervised, semi-supervised learning, linguistic analysis, textual analytics, and linked data, it can be presented as a model (Figure 3) that integrates situated learning and the analogous didactic model to have a better approach to the fundamental concepts of programming to be taught by the professor.

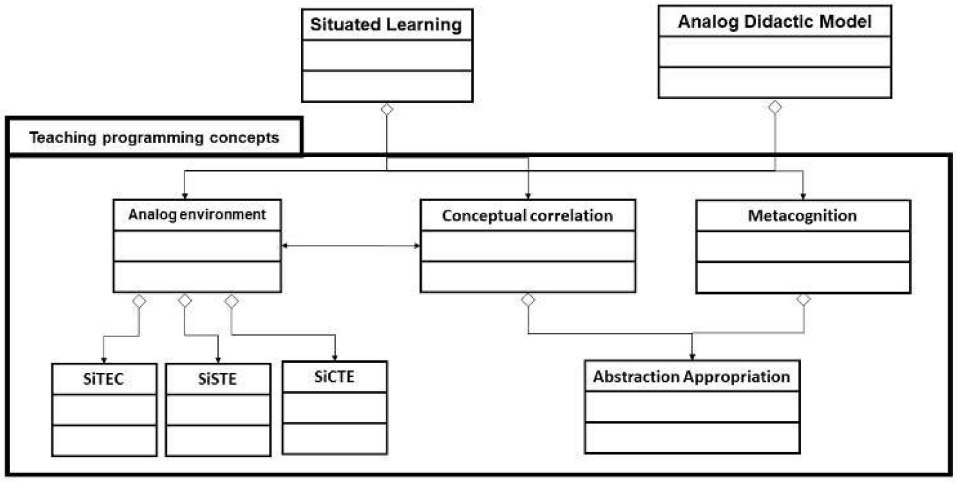

Figure 3. Model for teaching programming concepts.

Arias [79] states that the theory of situated learning establishes a relationship between the learner and the context, which is structured on a practical basis, therefore, to make the learning effective, the learner must be actively involved in actual instruction, that is, be part and product of the activity, the context, and the culture in which the instruction is developed and used [80].

On the other hand, the analogous didactic model constitutes a teaching strategy that implies the active construction, coming from the students, of the elements that constitute the base domain of the analogy [22].

The analogous environment refers to the real situation experienced by the students in their context; in this study, three environments were proposed:

1. Situated teaching environment in the classroom (SiTEC): When the situation is presented with physical elements in the classroom.

2. Situated and symbolic teaching environment (SiSTE): When it is observed indirectly (e.g., in simulated environments, immersive or digital environments, videos, among others).
3. Situated and Covert Teaching Environment (SiCTE): This occurs when the student who learns does so by imagining the situation.

In the conceptual correlation, the technical vocabulary is introduced, which is correlated with the options of the analogous environment to find meaning and understanding by comparing the concepts through the experiential analogy presented.

Finally, metacognition is the route of study of the professor's teaching process, which guides the planning, execution, and control of mental actions and operations of students, which have been oriented together with the aforementioned stages and there is an action to regulate the teaching of computer programming.

A key element of the model in Figure 3 is the analogy that the professor will use to teach a fundamental concept of programming. For the construction of an analogy, one must start from the concept to be taught, taking into consideration the three proposed analog environments (SiTEC, SiSTE, and SiCTE) and using experiential situations of the context where the learning takes place, so that the student can easily identify an analogy.

A proposal to build analogies and teach the concepts of programming fundamentals is to combine morphosyntactic elements with the most important findings of this study such as the inclusion of keywords from the terminology of computational examples, quantitative contexts, infinitive verbs, common nouns, and the sentiment analysis of sentences.

Therefore, Figure 4 presents the elements that an analogy may contain to teach these concepts through the use of sentences with elliptical or tacit subject that do not have the explicit subject in the grammatical structure, that is, they have the form of instructions and are categorized in the affirmative imperative grammatical mood.

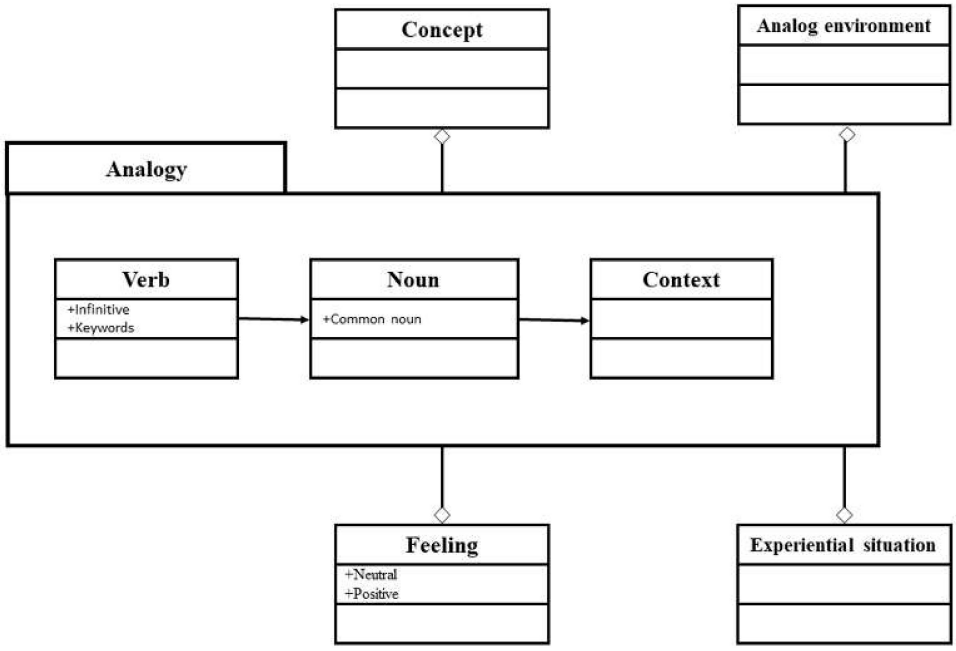

Figure 4. Elements of an analogy for programming concepts.

The findings presented in this study of supervised, unsupervised, semi-supervised learning, linguistic analysis, textual analytics, and linked data will allow us to select the most appropriate verbs, nouns and contexts for the concepts of input, output, conditional, and cycle, to later use computational thinking as a didactic strategy [81] in a CS1.

For example, according to the elements suggested in Figure 4, the following analogies can be used to teach the following concepts of programming fundamentals: enter a password in an ATM (input), obtain an invoice for the purchase of products (output), choose

a dish from the restaurant menu (conditional), and add the values of each product of a purchase (cycle).

Validation

The evaluation process of this study was carried out in two moments, first with professors of a CS1 and then with their students. For the first moment of evaluation, the positivist paradigm was used, with a quantitative approach, using the analytical empirical method, with a descriptive type of research and through a pre-experiment with pre-test and post-test.

In this experimental process, two professors participated from two universities in Colombia, CESMAG University (private) and University of Nariño (public), who belong to the Systems Engineering program in the subject called Introduction to Programming and Fundamentals of Programming, respectively, which is in the first academic semester of each curriculum.

Consequently, the experimental design proposed was G O_1 X O_2, where G is the experimental group formed by the two professors who guide the CS1 in each university. Likewise, O_1 was formed by the bank of examples of analogies that these professors were asked to write to explain the concepts of input, output, conditional and cycles, presenting a total of 14 examples for the course Introduction to Programming and 17 examples for Programming Fundamentals.

Then, the professors were given the experimental treatment X, which consisted of the model of the creation of analogies proposed in this research. Finally, they were asked to rebuild the analogies for each concept, according to the recommendations of the experimental treatment and whose result was O_2, obtaining the same number of initial examples raised by each professor.

When analyzing the analogies raised in O_1 with those obtained in O_2, it can be seen that the teachers considered elements such as the syntactic structure, the context, and the analogous environment. In addition, the first professor replaced six of the 14 examples initially proposed and the second professor changed five of the 17 examples. It was also observed that 100% of the initial analogies were restructured according to the syntactic recommendation to use a verb, a noun, and a context, incorporating the use of keywords as the main verb in each topic taught.

Similarly, all analogies in O_2 were shorter in length than those in O_1. For example, the analogy for teaching compound conditionals presented in O_1, whose wording was "when asking about the age of a student, you must establish if is of legal age or not", and in O_2, which was worded as follows: "Evaluate the age of a student and determine if is of legal age or not" being 37.2% smaller in extent, maintaining the context and adapting the syntactic structure.

In a second moment, the implementation of the analogies was evaluated according to the bank of examples obtained in O_2 with students of a CS1 and compared with the previous courses oriented by the same professors in which they used the analogies of O_1. In this case, the experiment had the same paradigm, approach, method, and type of research already mentioned with the professors, where the only difference was the research design, which was experimental with control groups and experimental with pre- and post-test.

In the experimental process, 154 students participated from the two universities already mentioned in San Juan de Pasto (Col) distributed in four groups and whose experimental design for CESMAG University was: G_1 O_3 X O_4 and G_2 O_5–O_6 and for The University of Nariño: G_3 O_7 X O_8 and G_4 O_9–O_{10}.

The groups G_1 and G_3 corresponded to the experimental groups of each university, while G_2 and G_4 were their control groups, respectively. In addition, X was the experimental treatment that consisted of using analogies with the model proposed in this research as a didactic strategy for teaching the fundamental concepts of programming in a CS1. In turn, O_3, O_5, O_7, and O_9 were the pre-tests implemented to evaluate the students' previous knowledge before facing the CS1. In addition, O_4, O_6, O_8, and O_{10} were the

post-tests performed at the end of the experimental treatment for both the experimental and control groups.

The G_1 group was formed by 45 students (89% men and 11% women, aged between 16 and 20 years, and two students who retook the course) in the course Introduction to Programming of the first semester of Systems Engineering at CESMAG University, carried out between February and May of 2021, to whom O_3 was implemented by means of a questionnaire to verify their previous knowledge in the concepts of input, output, conditional, and cycle. Then, during the course, the topics were explained through the experimental treatment X, and finally, the notes obtained in the academic evaluation process were considered as O_4 posttests.

Likewise, the G_2 control group was made up of 31 students (87% men and 13% women, aged between 17 and 22 years and a student who retook the course) who received the same course with the same professor during the months of August and November 2020 at the same university. This group was given the same initial questionnaire as the G_1 group as a pre-test (O_5), but no experimental treatment was applied (X), that is, the professor used the analogies proposed in O_1 and the notes obtained were considered as O_6.

On the other hand, G_3 was formed by 40 students (83% men and 17% women, aged between 16 and 25 years, and no retaking of the course) of the course Fundamentals of Programming of the first semester of the Systems Engineering program at the University of Nariño, to whom O_7 was also applied as a result of the previous knowledge, then, during the course, the topics were explained through the experimental treatment X, and finally, the notes obtained in the academic evaluation process were O_8.

In the same way, G_4 was the control group with 42 students (89% men and 11% women, aged between 16 and 23 years, and no retaking of the course) who received the same course with the same professor during May and September 2020 at the same university. This group was also given the same initial questionnaire as the G_3 group as a pre-test (O_9), but no experimental treatment (x) was applied and the grades obtained were O_{10}.

The evaluation design for the two experimental groups and the two control groups was done through two academic follow-up activities: A group workshop (45%) and an individual quiz (55%) for each of the topics proposed in this study. The averages of the grades obtained in the pre- and post-test for the four groups are shown in Table 8.

The marks obtained in the pre-tests of both the experimental and control groups were never higher than those obtained in the different posttests, concluding that students from both the public and private universities learn the fundamental concepts of programming in a CS1, therefore, it highlights the importance of this study. In addition, it is evident the difficulty presented in the concept of conditionals, obtaining positives results in the management of cycles and achieving an optimal appropriation in the handling of data input and output.

Finally, a statistical analysis was performed to determine by means of the student's probability T distribution [80], the difference that exists between the grades obtained by the experimental group and the control group in each university. Thus, in both G_1 and G_3, the statistical values t (3.46 and 2.06) were greater than both the critical value of t of a queue (1.67 in both cases) and the critical value for two queues (1.99 in both cases). In addition, the record for one and two P tails was less than 5% in each case, which concludes that the grades obtained by G_1 and G_3, compared to those of G_2 and G_4, in each theme were statistically different.

Therefore, the aforementioned statistical analysis demonstrates the impact that the analogies built under the model proposed here can decrease the complexity of the topics for a student in a CS1.

6. Conclusions and Further Work

Professors included keywords that are typical of the terminology for computational examples in the writing of the analogies used in a CS1. Additionally, most of the contexts

used in the analogies were of the quantitative type, where the numerical field has special importance.

In the sentence structure described by the professors when writing the analogy, the most significant grammatical element of value established by the NLP techniques was the verb, which at the same time was also the main attribute in the tasks of classification, association, and segmentation.

In turn, numerical examples, cooking recipes, and games were the most widely used categories as a reference for the construction of analogies that allow the student to abstract a fundamental concept of computer programming.

In addition, the results obtained in unsupervised and semi-supervised learning confirmed and complemented those obtained in supervised learning, the first being more specific and the second being more general.

It was also observed that the association rules generated in this study presented a higher level of confidence than those found in the classification rules. In addition, they allowed us to obtain patterns of greater specificity for each of the evaluated concepts.

Similarly, the results found by grouping ratified the rules obtained by association and classification in the four concepts.

On the other hand, the linguistic analysis showed us that despite having a wide lexical variety in the four concepts, due to the wide geographical area from which the data were taken, the professors used analogies in common contexts.

There was a high level of agreement among the results of the linguistic analysis, textual analysis, and data linked by each concept compared to the results obtained with the supervised and unsupervised analysis techniques.

For its part, the sentiment analysis showed that the analogies described by the professors had, on average, a neutral wording (46.6%) in the vast majority, followed by positive feelings (36.5%), and to a lesser degree negative (17.1%).

The design of an effective analogy to teach fundamental concepts in a CS1 must combine morphosyntactic elements with keywords in the form of verbs, and those must be typical of the terminology of computational examples. In addition, they must be in quantitative contexts and accompanied by common nouns and with neutral or positive phrases.

For future work, we will design a recommendation system that will guide the professor in the approach of an analogy to address a fundamental concept of a CS1 and develop a visualization tool that supports the process of abstraction in computing environments.

Author Contributions: Conceptualization, J.A.J.T., C.A.C., and M.O.; Methodology, J.A.J.T., C.A.C., and M.O.; Validation, J.A.J.T., C.A.C., and M.O.; Formal analysis, J.A.J.T., C.A.C., and M.O.; Investigation, J.A.J.T., C.A.C., and M.O.; Resources, J.A.J.T., C.A.C., and M.O.; Data curation, J.A.J.T., C.A.C., and M.O.; Writing—original draft preparation, J.A.J.T., C.A.C., and M.O.; Writing—review and editing, J.A.J.T., C.A.C., and M.O. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data presented in this study are available on request from the corresponding author.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. López Reguera, J.; Hernández Rivas, C.; Farran Leiva, Y. An Automatic Evaluation Platform with an Effective Methodology for Teaching/Learning in Computer Programming. *Ingeniare Rev. Chil. Ing.* **2011**, *19*, 265–277. [CrossRef]
2. Depetris, B.; Mallea, D.A.; Pendenti, H.; Tejero, G.; Prisching, G. Teaching and Learning Programming and Concurrent Programming with DaVinci. In Proceedings of the X Congress of Technology in Education and Education in Technology, Corrientes, Argentina, 11–12 June 2015; pp. 194–202.

3. Silva, G.; Arjona, P.; Castillo, F. More Time or Better Tools? A Large-Scale Retrospective Comparison of Pedagogical Approaches to Teach Programming. *IEEE Trans. Educ.* **2016**, *59*, 274–281. [CrossRef]
4. Sánchez, S.; García, M.Á.; Lacave, C.; Molina, A.I.; González, C.; Vallejo, D.; Redondo, M.Á.; Sanchez, E.S.; Gmarin, M.; Lacave, C.; et al. Applying Mixed Reality Techniques for the Visualization of Programs and Algorithms in a Programming Learning Environment. In Proceedings of the eLmL 2018: The Tenth International Conference on Mobile, Hybrid, and On-line Learning Applying, Rome, Italy, 25–29 March 2018; pp. 84–89.
5. Lacave, C.; Garcia, M.A.; Molina, A.I.; Sanchez, S.; Redondo, M.A.; Ortega, M. COLLECE-2.0: A Real-Time Collaborative Programming System on Eclipse. In Proceedings of the 2019 International Symposium on Computers in Education (SIIE), Tomar, Portugal, 21–23 November 2019; pp. 1–6. [CrossRef]
6. Redondo, M.Á.; Ortega, M. Colecce 2.0. Available online: http://blog.uclm.es/grupochico/proyecto-iapro/collece-2-0/ (accessed on 27 March 2021).
7. Sukamto, R.; Megasari, R. Analogy Mapping for Different Learning Style of Learners in Programming. In Proceedings of the 2017 3rd International Conference on Science in Information Technology (ICSITech), Bandung, Indonesia, 25–26 October 2017; pp. 626–631. [CrossRef]
8. Strunz, K.; Louie, H. Cache Energy Control for Storage: Power System Integration and Education Based on Analogies Derived From Computer Engineering. *IEEE Trans. Power Syst.* **2009**, *24*, 12–19. [CrossRef]
9. Plappally, A. The effect of joint role of creative analogy and concept-in-context map on the learning interest and performance of first year mechanical engineering undergraduates. In Proceedings of the 2016 IEEE 8th International Conference on Engineering Education (ICEED), Kuala Lumpur, Malaysia, 7–8 December 2016; pp. 126–130. [CrossRef]
10. Stockdill, A.; Raggi, D.; Jamnik, M.; Garcia, G.; Sutherland, H.; Cheng, P.; Sarkar, A. Correspondence-Based Analogies for Choosing Problem Representations. In Proceedings of the 2020 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC), Dunedin, New Zealand, 10–14 August 2020; pp. 1–5. [CrossRef]
11. Chaves, C.; Rosero, M.M. Teaching Model and Its Relationship with Metacognitive Processes in Systems Programming. *Rev. Educ. Ing.* **2014**, *9*, 1–12. [CrossRef]
12. Jiménez, J.; Collazos, C.; Hurtado, J.; Pantoja, W. Collaborative Strategy in Three-Dimensional Environments as a Didactic Strategy for Learning Iterative Structures in Computational Programming. *Investig. IRE Cienci. Soc. Hum.* **2015**, *6*, 80–92. [CrossRef]
13. Jiménez-Toledo, J.A.; Collazos, C.; Revelo-Sánchez, O. Considerations in the Teaching-Learning Processes for a First Course in Computer Programming: A Systematic Review of the Literature. *TecnoLógicas* **2019**, *22*, 83–117. [CrossRef]
14. Hundhausen, C.D.; Douglas, S.A.; Stasko, J.T. A Metastudy of Algorithm Visualization Effectiveness. *J. Vis. Lang. Comput.* **2002**, *13*, 259–290. [CrossRef]
15. Ruiz, F.J.; Luciano, C. Relating Relationships as a Functional Analytical Model of Analogy and Metaphor. *Acta Comport.* **2012**, *20*, 3–29.
16. Glynn, S.; Brillan, B.; Semrud Clikman, M.; Muth, K. Analogical Reasoning and Problem Solving in Science Textbooks. In *Handbook of Creativity*; PlenumPress: NewYork, NY, USA, 1989; pp. 383–398.
17. Haaparanta, L. The Analogy Theory of Thinking. *Dialectica* **1992**, *46*, 169–183. [CrossRef]
18. Perelman, C. *Analogie et Metaphore En Science, Poesie et Philosophie*; Presses Universitaires de Bruxelles: Bruxelles, Belgium, 1970.
19. Oliva, J.M. Actividades Para La Enseñanza/Learning chemistry through analogies. *Eureka Mag. Sci. Teach. Dissem.* **2006**, *3*, 104–114. [CrossRef]
20. Gilbert, J.K. *Multiple Representations in Chemical Education*; Gilbert, J.K., Treagust, D., Eds.; Models and Modeling in Science Education; Springer: Dordrecht, The Netherlands, 2009; Volume 4. [CrossRef]
21. Harrison, A.G.; Treagust, D.F. A Typology of School Science Models. *Int. J. Sci. Educ.* **2000**, *22*, 1011–1026. [CrossRef]
22. Galagovsky, L.; Adúriz-Bravo, A. Models and Analogies in the Teaching of Natural Sciences. The Concept of Analogical Didactic Model. *Enseñ. Cienc.* **2001**, *19*, 231–242.
23. Fernández González, J.; González González, B.M.; Moreno Jiménez, T. Considerations about Research in Analogies. *Estud. Front.* **2004**, *5*, 79–105. [CrossRef]
24. Malachías, M.E.I.; Borges dos Santos, D. Critical Meaningful Learning through the Explanatory Proposition of Analogies Through the Analog Didactic Model (MDA). *Electron. J. Res. Sci. Educ.* **2013**, *8*, 21–33.
25. Wang, D.; Su, J.; Yu, H. Feature Extraction and Analysis of Natural Language Processing for Deep Learning English Language. *IEEE Access* **2020**, *8*, 46335–46345. [CrossRef]
26. Ahmet, C. *Artificial Intelligence: How Advance Machine Learning Will Shape the Future of Our Word*; Independently published, 2018. Available online: https://www.amazon.com/Artificial-Intelligence-Advanced-Machine-Learning/dp/1790129753 (accessed on 27 March 2021).
27. Sharda, R.; Delen, D.; Turban, E. *Pearson Etext Analytics, Data Science, & Artificial Intelligence*, 11th ed.; Pearson Education: London, UK, 2019.
28. Timarán, S.; Hernández, I.; Caicedo, S.; Hidalgo, A.; Alvarado, J. *Discovery of Academic Performance Patterns with Decision Trees in Generic Professional Training Competencies*, 1st ed.; Universidad Cooperativa de Colombia: Bogotá, Colombia, 2016. [CrossRef]
29. Gomes, R.P.; Ribeiro, V.G.; Corrêa, Y.; Zabadal, J.R.S. Aplicação de Revisão Sistemática Com Suporte de Mineração de Dados e de Textos: O Caso Do Periódico Design Studies. *Em. Quest.* **2019**, *25*, 156–183. [CrossRef]

30. Arce, D.; Lima, F.; Orellana Cordero, M.P.; Ortega, J.; Sellers, C.; Ortega, P. Discovering Behavioral Patterns among Air Pollutants: A Data Mining Approach. *Enfoque UTE* **2018**, *9*, 168–179. [CrossRef]
31. Tan, P.-N.; Steinbach, M.; Karpatne, A.; Kumar, V. *Introduction to Data Mining*; Pearson: Edinburgo, UK, 2019.
32. Jordan, M.I.; Mitchell, T.M. Machine Learning: Trends, Perspectives, and Prospects. *Science* **2015**, *349*, 255–260. [CrossRef]
33. Xue, M.; Yuan, C.; Wu, H.; Zhang, Y.; Liu, W. Machine Learning Security: Threats, Countermeasures, and Evaluations. *IEEE Access* **2020**, *8*, 74720–74742. [CrossRef]
34. Qian, G.; Li, Z.; He, C.; Li, X.; Ding, X. Power Allocation Schemes Based on Deep Learning for Distributed Antenna Systems. *IEEE Access* **2020**, *8*, 31245–31253. [CrossRef]
35. Ledesma, S.; Ibarra-Manzano, M.; Cabal-Yepez, E.; Almanza-Ojeda, D.; Avina-Cervantes, J. Analysis of Data Sets with Learning Conflicts for Machine Learning. *IEEE Access* **2018**, *6*, 45062–45070. [CrossRef]
36. Bo, L.; Wang, L.; Jiao, L. Feature Scaling for Kernel Fisher Discriminant Analysis Using Leave-One-Out Cross Validation. *Neural Comput.* **2006**, *18*, 961–978. [CrossRef] [PubMed]
37. Jin, B.; Jing, Z.; Zhao, H. Incremental and Decremental Extreme Learning Machine Based on Generalized Inverse. *IEEE Access* **2017**, *5*, 20852–20865. [CrossRef]
38. Russell, S.; Norvig, P. *Artificial Intelligence: A Modern Approach*, 3rd ed.; Norvig, P., Ed.; Pearson: London, UK, 2016.
39. Marsland, S. *Machine Learning: An Algorithmic Perspective*, 2nd ed.; CRC Press: Boca Raton, FL, USA, 2015.
40. Godoy Viera, Á.F. Machine Learning Techniques Used for Text Mining. *Investig. Bibl. Arch. Bibl. Inf.* **2017**, *31*, 103. [CrossRef]
41. Verhaar, P. *Text and Data Mining: The Theory and Practice of Using TDM for Scholarship in the Humanities*; Facet Publishing: London, UK, 2020.
42. Alvarez Fernández, N.; Comin, X.J.; Merino Arranz, D. *Técnicas de Machine Learning y Desarrollo de Modelos Predictivos Aplicados a la Antropología Forense*; Universidad Oberta de Catalunya: Barcelona, Spain, 2018.
43. LeCun, Y.; Bengio, Y.; Hinton, G. Deep Learning. *Nature* **2015**, *521*, 436–444. [CrossRef] [PubMed]
44. Zhong, G.; Zhang, K.; Wei, H.; Zheng, Y.; Dong, J. Marginal Deep Architecture: Stacking Feature Learning Modules to Build Deep Learning Models. *IEEE Access* **2019**, *7*, 30220–30233. [CrossRef]
45. Arce García, S.; Menéndez Menéndez, M.I. Applications of Statistics to Framing and Text Mining in Communication Studies. *Inf. Cult. Soc.* **2018**, 61–70. [CrossRef]
46. Lin, F.; Hao, D.; Liao, D. Automatic Content Analysis of Media Framing by Text Mining Techniques. In Proceedings of the 2016 49th Hawaii International Conference on System Sciences (HICSS), Koloa, HI, USA, 5–8 January 2016; pp. 2770–2779. [CrossRef]
47. Kwartler, T. *Text Mining in Practice with R*, 1st ed.; Jhon Wiley & Sons Ltd.: Oxford, UK, 2017.
48. Ortíz, Z. Trends and Challenges for Information Science in Today's World. *Cienci. Inf.* **2019**, *9*, 196–208. [CrossRef]
49. Mandujano, S. Analysis and Trends of Photo-Trapping in Mexico: Text Mining in R. *Therya* **2019**, *10*, 25–32. [CrossRef]
50. Rashid, J.; Adnan Shah, S.M.; Irtaza, A.; Mahmood, T.; Nisar, M.W.; Shafiq, M.; Gardezi, A. Topic Modeling Technique for Text Mining Over Biomedical Text Corpora Through Hybrid Inverse Documents Frequency and Fuzzy K-Means Clustering. *IEEE Access* **2019**, *7*, 146070–146080. [CrossRef]
51. Kim, D.; Lee, J.; So, C.H.; Jeon, H.; Jeong, M.; Choi, Y.; Yoon, W.; Sung, M.; Kang, J. A Neural Named Entity Recognition and Multi-Type Normalization Tool for Biomedical Text Mining. *IEEE Access* **2019**, *7*, 73729–73740. [CrossRef]
52. Zhang, L.; Zhu, G.; Zhang, S.; Zhan, X.; Wang, J.; Meng, W.; Fang, X.; Wang, P. Assessment of Career Adaptability: Combining Text Mining and Item Response Theory Method. *IEEE Access* **2019**, *7*, 125893–125908. [CrossRef]
53. Shi, L.; Jianping, C.; Jie, X. Prospecting Information Extraction by Text Mining Based on Convolutional Neural Networks—A Case Study of the Lala Copper Deposit, China. *IEEE Access* **2018**, *6*, 52286–52297. [CrossRef]
54. Jia, S.; Wu, B. Incorporating LDA Based Text Mining Method to Explore New Energy Vehicles in China. *IEEE Access* **2018**, *6*, 64596–64602. [CrossRef]
55. Verma, V.K.; Ranjan, M.; Mishra, P. Text Mining and Information Professionals: Role, Issues and Challenges. In Proceedings of the 2015 4th International Symposium on Emerging Trends and Technologies in Libraries and Information Services, Noida, India, 6–8 January 2015; pp. 133–137. [CrossRef]
56. Sukanya, M.; Biruntha, S. Techniques on Text Mining. In Proceedings of the 2012 IEEE International Conference on Advanced Communication Control and Computing Technologies (ICACCCT), Ramanathapuram, India, 23–25 August 2012; pp. 269–271. [CrossRef]
57. Miner, G.; Nisbet, B.; Elder, J.; Fast, A.; Delen, D.; Hill, T. *Practical Text Mining and Statistical Analysis for Non-Structured Text Data Applications*; Academic Press Elsevier: Oxford, UK, 2012.
58. Uysal, A.; Gunal, S. The Impact of Preprocessing on Text Classification. *Inf. Process. Manag.* **2014**, *50*, 104–112. [CrossRef]
59. Raschka, S.; Mirjalili, V. *Python Machine Learning*, 2nd ed.; Packt Publishing: Birmingham, UK, 2017.
60. Lobur, M.; Romanyuk, A.; Romanyshyn, M. Using NLTK for Educational and Scientific Purposes. *IEEE Xplore* **2011**, *1*, 43.
61. Xu, L.; Sun, S.; Wang, Q. Text Similarity Algorithm Based on Semantic Vector Space Model. In Proceedings of the 2016 IEEE/ACIS 15th International Conference on Computer and Information Science (ICIS), Okayama, Japan, 26–29 June 2016; pp. 1–4. [CrossRef]
62. Torres, C.; Arco, L. Textual Representation in Semantic Vector Space. *Rev. Cuba. Cienci. Inf.* **2016**, *10*, 148–180.
63. Kim, J.-K.; Tur, G.; Celikyilmaz, A.; Cao, B.; Wang, Y.-Y. Intent Detection Using Semantically Enriched Word Embeddings. In Proceedings of the 2016 IEEE Spoken Language Technology Workshop (SLT), San Diego, CA, USA, 13–16 December 2016; pp. 414–419. [CrossRef]

64. Sano, N. Synthetic Data by Principal Component Analysis. In Proceedings of the 2020 International Conference on Data Mining Workshops (ICDMW), Sorrento, Italy, 17–20 November 2020; pp. 101–105. [CrossRef]
65. Yadav, S.; Shukla, S. Analysis of K-Fold Cross-Validation over Hold-Out Validation on Colossal Datasets for Quality Classification. In Proceedings of the 2016 IEEE 6th International Conference on Advanced Computing (IACC), Bhimavaram, India, 27–28 February 2016; pp. 78–83. [CrossRef]
66. Sedaghat, N.; Fathy, M.; Modarressi, M.H.; Shojaie, A. Combining Supervised and Unsupervised Learning for Improved MiRNA Target Prediction. *IEEE/ACM Trans. Comput. Biol. Bioinform.* **2018**. [CrossRef] [PubMed]
67. Cedeno, D.; Vargas, M. Application of Machine Learning with Supervised Classification Algorithms: In the Context of Health. In Proceedings of the 2019 7th International Engineering, Sciences and Technology Conference (IESTEC), Panama, 9–11 October 2019; pp. 613–618. [CrossRef]
68. Nijhawan, R.; Srivastava, I.; Shukla, P. Land Cover Classification Using Super-Vised and Unsupervised Learning Techniques. In Proceedings of the 2017 International Conference on Computational Intelligence in Data Science (ICCIDS), Chennai, India, 2–3 June 2017; pp. 1–6. [CrossRef]
69. Karacali, B. Improved Quasi-Supervised Learning by Expectation-Maximization. In Proceedings of the 2013 21st Signal Processing and Communications Applications Conference (SIU), Haspolat, Turkey, 24–26 April 2013; pp. 1–4. [CrossRef]
70. Lytvyn, V.; Vysotska, V.; Veres, O.; Rishnyak, I.; Rishnyak, H. Content Linguistic Analysis Methods for Textual Documents Classification. In Proceedings of the 2016 XIth International Scientific and Technical Conference Computer Sciences and Information Technologies (CSIT), Lviv, Ukraine, 6–10 September 2016; pp. 190–192. [CrossRef]
71. Weir, G.; Owoeye, K.; Oberacker, A.; Alshahrani, H. Cloud-Based Textual Analysis as a Basis for Document Classification. In Proceedings of the 2018 International Conference on High Performance Computing & Simulation (HPCS), Orleans, France, 16–20 July 2018; pp. 672–676. [CrossRef]
72. Yun, D.; Liu, W.; Wu, C.Q.; Rao, N.S.V.; Kettimuthu, R. Performance Prediction of Big Data Transfer Through Experimental Analysis and Machine Learning. In Proceedings of the 2020 IFIP Networking Conference (Networking), Paris, France, 22–26 June 2020; pp. 181–189.
73. Gamallo, P.; García, M. *Methods on Natural Language Processing for Information Retrieval*; Universidade de Santiago de Compostela: Santiago, Chile, 2012.
74. Pérez-Guadarramas, Y.; Rodríguez-Blanco, A.; Simón-Cuevas, A.; Hojas-Mazo, W.; Olivas, J.Á. Combining Lexical—Syntactic Patterns and Topic Analysis for Automatic Keyphrase Extraction from Texts. *Proces. Leng. Nat.* **2017**, *59*, 39–46.
75. Rodriguez-Blanco, A.; Simon-Cuevas, A.; Hojas-Mazo, W.; Perea-Ortega, J. Extraction of Linked Data from Unstructured Information Applying NLP Techniques and Ontologies. *CEUR Workshop Proc.* **2016**, *1797*, 80–91.
76. Perez-Marin, D.; Hijon-Neira, R.; Martin-Lope, M. A Methodology Proposal Based on Metaphors to Teach Programming to Children. *IEEE Rev. Iberoam. Tecnol. Aprendiz.* **2018**, *13*, 46–53. [CrossRef]
77. Pérez-Marín, D.; Hijón-Neira, R.; Bacelo, A.; Pizarro, C. Can Computational Thinking Be Improved by Using a Methodology Based on Metaphors and Scratch to Teach Computer Programming to Children? *Comput. Hum. Behav.* **2020**, *105*, 105849. [CrossRef]
78. Chibaya, C. A Metaphor-Based Approach for Introducing Programming Concepts. In Proceedings of the 2019 International Multidisciplinary Information Technology and Engineering Conference (IMITEC), Vanderbijlpark, Kimberley, South Africa, 21–22 November 2019; pp. 1–8. [CrossRef]
79. Arias, L. Situated Learning and Cognitive Development. Available online: https://docplayer.es/27825700-El-aprendizaje-situado-y-el-desarrollo-cognitivo-comparacion-entre-las-teorias-aprendizaje-situado-y-desarrollo-cognitivo-de-brunner.html (accessed on 27 March 2021).
80. Sanchez, R. T-Student, uses and abuses. *Rev. Mex. Cardiol.* **2015**, *26*, 59–61.
81. Ramos, X.; Jiménez, J. *Methodological Proposal for the Development of Computational Thinking*, 1st ed.; CESMAG University: Pasto, Colombia, 2020; Available online: http://repositorio.unicesmag.edu.co:8080/xmlui/handle/123456789/126 (accessed on 27 March 2021).