





Article

Recommending Learning Objects with Arguments and Explanations

Stella Heras ^{1,†,‡} , Javier Palanca ^{1,†} , Paula Rodriguez ^{2,‡}, Néstor Duque-Méndez ^{3,‡} 
and Vicente Julian ^{1,†,‡,*} 

¹ Valencian Research Institute for Artificial Intelligence (VRAIN), Universitat Politècnica de València, Camino de Vera s/n, 46022 Valencia, Spain; sheras@dsic.upv.es (S.H.); jpalanca@dsic.upv.es (J.P.)

² Departamento de Sistemas, Instituto Tecnológico Metropolitano, Medellín 050034, Colombia; parodriguezma@itm.edu.co

³ Departamento de Informática y Computación, Universidad Nacional de Colombia, Manizales 17003, Colombia; ndduqueme@unal.edu.co

* Correspondence: vjulian@upv.es

† Current address: Valencian Research Institute for Artificial Intelligence, Universitat Politècnica de València, Camino de Vera s/n, 46022 Valencia, Spain.

‡ These authors contributed equally to this work.

Received: 23 March 2020; Accepted: 7 May 2020; Published: 12 May 2020



Abstract: The massive presence of online learning resources leads many students to have more information than they can consume efficiently. Therefore, students do not always find adaptive learning material for their needs and preferences. In this paper, we present a Conversational Educational Recommender System (C-ERS), which helps students in the process of finding the more appropriated learning resources considering their learning objectives and profile. The recommendation process is based on an argumentation-based approach that selects the learning objects that allow a greater number of arguments to be generated to justify their suitability. Our system includes a simple and intuitive communication interface with the user that provides an explanation to any recommendation. This allows the user to interact with the system and accept or reject the recommendations, providing reasons for such behavior. In this way, the user is able to inspect the system's operation and understand the recommendations, while the system is able to elicit the actual preferences of the user. The system has been tested online with a real group of undergraduate students in the Universidad Nacional de Colombia, showing promising results.

Keywords: educational recommender systems; explanations; argumentation

1. Introduction

Nowadays, the increasing number of learning resources available online has encouraged the massive development of Technology Enhanced Learning systems (TEL) [1]. TEL systems make education accessible to anyone, regardless of where they live. Using this technology, many universities now offer Massive Online Open Courses (MOOCs) and courses in a flipped classroom format [2]. This new teaching methodology is based on the use of TEL systems, which offer different types of learning objects (LO, e.g., videos, tutorials) that students can consume online in a non-classroom setting, and thus classroom sessions can be used to resolve exercises and doubts, group work, and other activities that require the direct presence of the teacher. However, this massive availability of LOs brings an overload problem, since students must be able to select the most suitable LOs for their learning objectives from a large number of available LOs.

This opens new challenges for the research community on recommender systems, where Educational Recommender Systems (ERS) is now a trending topic. By using ERS, students

can efficiently get those LOs that are most appropriate to their educational level, learning objectives, and learning style. The student's learning style determines which LOs are more adequate for particular types of students. For instance, the widely applied VARK model (<http://vark-learn.com/>) identifies four distinct learning styles (i.e., Visual, Auditory, Reading, and Kinesthetic) and uses these dimensions to determine the appropriateness of specific types of online learning resources. For instance, visual learners rely more on diagrams, charts or graphs, while kinesthetic students learn best from simulations, videos and movies of 'real' things. Therefore, it is important to consider students' learning styles while developing online learning systems [3] and, consequently, educational recommender systems.

In [4], an ERS was presented that helps undergraduate students to find the LOs that are more suitable for them, taking into account their profile (level of education, preferred language, topic and format, learning style, and other personal information), learning practices (LOs that they already used), and similarity with other students. The system is a hybrid recommendation engine that combines computational argumentation with other several recommendation techniques (collaborative, content-based, and knowledge-based). This technique, which selects those LOs whose recommendation can be better supported by arguments, demonstrated being the most effective for this application. A detailed description of the technique and the different recommender algorithms can be found in [5].

The system offers a straightforward way of generating textual explanations from arguments. These arguments capture common inferences of human reasoning. Thus, while arguments are formal representations for the justification of each recommendation (as elements of an argumentation theory), explanations are textual representations of these arguments (as elements of the user interface). These explanations are used to convince students to accept the recommendations of the system and make use of the LO recommended.

However, online education is a tricky domain where learners' preferences can be highly time-variant. In addition, some students may have preferences that contradict the recommendations of learning theories (for example, because of their usual study practices). These acquired study habits are difficult to modify and students can quickly leave the ERS if they do not find it useful and effective for their learning objectives. Learners can even dynamically change their learning preferences or exhibit ad-hoc ones for specific topics or LOs.

Conversational recommender systems are particularly effective for preference elicitation through question-asking and recommendation critiquing [6,7]. Therefore, in this paper, a new conversational version of our system, which we call C-ERS (Conversational Educational Recommender System), is presented. This system can interact with users, allowing them to express their opinions on the recommendations, to specify their current preferences, and to correct the system's wrong assumptions. Its combination of argumentation theory, explanation generation, and conversational recommendation provides C-ERS with powerful tools to generate effective recommendations and engage students with the system.

To evaluate the system, it has been performed a live-user trial with a real group of undergraduate students of the Universidad Nacional de Colombia, obtaining promising results.

2. Related Work

It has been acknowledged that online recommender approaches suffer from the cost of retraining the model, are built to optimize online performance which does not necessarily match online user behavior, and fail to capture preference drifts [7]. These challenges have boosted research towards the development of continuously learning systems, such as conversational recommender systems.

Conversational Recommender Systems are usually used to quickly establish preferences for users when we don't know too much about the one we are recommending. This is especially helpful to address the cold-start recommendation problem. The problem of eliciting user preferences can be addressed through different methods like interviews-based strategies, asking the user to rate some items, active learning, interactivity, or utilizing feedback in an online setting [7–9]. Interactive recommenders give the user a more active role so that the recommender system can improve their

recommendations online. There are different works that use this approach, like critique-based [10], constraint-based [11], or dialogical recommenders [12].

Our system differs from critiquing-based recommenders since users do not critique items explicitly, but the way of reasoning that the system has followed to propose those items. In addition, our system is less affected by the cold-start problem, since recommendations are based on learning theories and argumentation inferences that also apply to new users once they have logged in the system and their learning profile is established. Furthermore, all these approaches have some kind of interactivity with the user to try to learn more from him/her. However, this work will not only try to learn and modify what the system knows about the user during the conversation, but in addition it will also try to persuade the user towards the best choice using arguments and explanations.

In [13], the authors present a survey where they compare 24 interactive recommender systems from the point of view of visualization, which is an effective approach to facilitate human perception and understanding of what is being recommended to the user. The authors have clustered the 24 analyzed recommenders into different clusters that focus on a feature of the recommender (transparency, justification, controllability, etc.). The feature most closely related to this work is the ability to generate justifications. Supporting justifications is usually used to explain the recommendations by comparing item tags with user preferences. Two examples are *Tagsplanation* [14] and *MovieExplain* [15]. In [16,17], there are more examples of how to use justifications for recommender systems. The way recommendations are presented to the user influence in the explanation attributes (or metrics) that are used. These attributes are: *transparency* (explain how the system worked), *scrutability* (which allows users to tell the system it is wrong), *trust* (to increase user's confidence in the system), *effectiveness* (to help users make good decisions), *efficiency* (to help users make decisions faster), *satisfaction* (to help the user to enjoy the use of the system), and *persuasiveness* (to convince users to choose the recommended item). This work focuses in the persuasiveness skills of the system to convince users by means of arguments generated during the recommender process.

The literature on persuasion systems reports that users find suggestions from sources that have proven to be reliable more credible [18]. Thus, the more credible a recommender system is, the more likely are its recommendations to be accepted. However, in [19], the authors analyze how the credibility of the system can be captured by a proper selection of its features. There are different approaches to follow, but *explanations* (i.e., arguments) is an efficient way to increase the system credibility, since they can clarify the system recommendation (the good intentions and efforts it does to provide the user the best recommendation) to users [20]. As has been presented above, some attributes or metrics help the user to trust more on recommendations. That is, they better accept recommendations if they understand how they were generated by inspecting how the system works (transparency), or if they can understand and evaluate why recommendations can be suitable to them (education). Furthermore, it has been shown that even if the recommended items are already known, users still prefer the system to provide explanations [21]. Explanations may also avoid bad choices. As explained in [17], the possibility of generating a certain type (style) of explanations depends largely on the algorithm being used and its ability to produce the information needed to generate these explanations. The styles of explanations that C-ERS can generate are directly conditioned by the type of knowledge that the rules of the system handle (collaborative, content-based, and knowledge-based).

Research in ERS has been very active in recent years [22–26]. Providing recommendation techniques with insights and explanations to learners and educators has become an important research line [1]. Argumentation theory and their tools have shown many successes in educational domains, especially regarding the training of critical thinking for law students [27]. Recommender systems combined with computational argumentation techniques have reported successful applications to suggest music [28], news [29], movies [30], or content-based web search [31]. Among the proposed systems, closer similarities exist between our approach and those proposed in [32], a recommender system that uses defeasible logic programming to recommend movies, and [33], a products recommender system based on the Toulmin's argumentation theory. In the former,

the authors resolve possible attacks between arguments by defining a pre-established criterion of preference between rules. However, our system uses a probabilistic method to calculate the probability of one argument prevailing over another, which makes the system more adaptable. In the latter, the system allows for generating arguments to support the recommendation of specific items (cameras) and their features. However, the Toulmin's model only allows for generate claims in favor or against a specific conclusion (the recommendation of the item), while our system allows for generating different explanation styles from content-based, knowledge-based, and collaborative information sources and that allow for critiquing the explanation itself and not only the recommendation.

Although the impact of explanations on the performance of recommender systems has been proved [19], few research has been focused on the domain of educational recommender systems. Designing recommender systems for TEL environments presents several challenges [34]. For instance, each student learns following a different learning process, with a variety of learning methods and tools. Our work involves a contribution in these areas by proposing a conversational ERS provided with an argumentation framework that makes the system able to engage in a conversation with students to help them to understand the underlying logic of the system, to elicit drifts in their preferences, and to provide effective recommendations.

3. Conversational Educational Recommender System (C-ERS)

In this section, the recommendation and conversational functionalities of the argumentation-based C-ERS is presented. The proposed system uses expert knowledge encoded in defeasible rules to match user preferences with available LOs and infer suitable recommendations. Then, the system interacts with the user to validate its assumptions by providing explanations. We refer the reader to [5] for specific details on the recommendation algorithms and an evaluation on the performance of different recommendation techniques in this domain.

3.1. Recommendation Process

C-ERS follows a hybrid recommendation technique that uses LOs *metadata* and information extracted from the *student's profiles* to compute the recommendations. Concretely, to represent LOs the proposed system follows the *IEEE-LOM* standard (1484.12.1-2002—IEEE Standard for Learning Object Metadata: <https://standards.ieee.org/findstds/standard/1484.12.1-2002.html>). These profiles provide personalized information about the students, their interactivity level, the language they prefer, some preferences about the format, their learning style (which can be an auditory learning style, a kinaesthetic learning style, a reader learning style, or a visual learning style), and their usage history (see Section 4). The learning styles have been obtained using the Spanish adaptation of the VARK questionnaire, available at [35].

This technique combines three types of recommender systems: one based on the content of the items being recommended, one based on comparing user's actions (called collaborative), and one based on the knowledge of the domain [36]. This technique models this expert knowledge in a logic program. The proposal of this work is to use a formalism based on defeasible argumentation (which is founded on logic programming, similar to *DeLP*[37]) to implement the logic of the recommendation system and build arguments to endorse the recommendation of selected Learning Objects. Thus, our C-ERS gives the users the Learning Objects that have better support by a greater number of arguments.

A defeasible logic program $P = (\Pi, \Delta)$ is modeled by strict knowledge (Π) and defeasible knowledge (Δ) about the domain of the application. In C-ERS, Π represents a set of *facts*. These facts are strict inference rules with an empty body. In the example logic program of Table 1, lines 1 to 7 represent facts. The fact *user_type(alice, visual)* represents a student named 'alice' with a visual learning style. This learning style means that the student has a preference for learning objects with images or graphs such as png or jpeg, slides such as ppt, etc.

Table 1. Example logic program.

1:	<code>user_type(alice, visual)</code>
2:	<code>resource_type(LO₁, slides)</code>
3:	<code>structure(LO₁, atomic)</code>
4:	<code>state(LO₁, final)</code>
5:	<code>similarity(alice, paul) > α</code>
6:	<code>similarity(LO₂, LO) > β</code>
7:	<code>vote(paul, LO₁) ≥ 4</code>
8:	<code>interactivity_type(LO₁, low) ← resource_type(LO₁, slides)</code>
9:	<code>appropriate_resource(alice, LO₁) ← user_type(alice, visual) ∧ resource_type(LO₁, slides)</code>
10:	<code>appropriate_interactivity(alice, LO₁) ← user_type(alice, visual) ∧ interactivity_type(LO₁, low)</code>
11:	<code>educationally_appropriate(alice, LO₁) ← appropriate_resource(alice, LO₁) ∧ appropriate_interactivity(alice, LO₁)</code>
12:	<code>generally_appropriate(LO₁) ← structure(LO₁, atomic) ∧ state(LO₁, final)</code>
13:	<code>recommend(alice, LO₁) ← educationally_appropriate(alice, LO₁) ∧ generally_appropriate(LO₁)</code>
14:	<code>recommend(alice, LO₁) ← similarity(alice, paul) > α ∧ vote(paul, LO₁) ≥ 4</code>
15:	<code>recommend(alice, LO₂) ← similarity(LO₂, LO) > β ∧ vote(alice, LO) ≥ 4</code>

Furthermore, the Δ set depicts defeasible rules of the form $P \leftarrow Q_1 \wedge \dots \wedge Q_k$. The above rules encode the defeasible inference that literals $Q_1 \wedge \dots \wedge Q_k$ can supply arguments to support the belief in P . Several types of defeasible rules constitute the logic behind each of the recommendations of the system. A summary of these rules and their associated explanations (explanations are commented in Section 3.2) are shown in Table 2 (Due to readability reasons, the full set of rules are not provided).

Table 2. C-ERS Defeasible rules and Explanations.

General Rules	
<hr/>	
<i>G1:</i>	<code>recommend(user, LO) ← cost(LO) = 0</code>
Explanation: 'This LO may interest you, since it is for free'	
Responses:	
+RG1: 'Accept'	
~RG1: 'Not sure. I don't care about the cost'	
-RG1: 'Reject. I don't like it'	
<hr/>	
<i>G2:</i>	<code>recommend(user, LO) ← quality_metric(LO) ≥ 0.7</code>
Explanation: 'This LO may interest you, since its quality is high'	
Responses:	
+RG2: 'Accept'	
~RG2: 'Not sure. I don't care about the quality'	
-RG2: 'Reject. I don't like it'	
<hr/>	
Content-based Rules	
<hr/>	
<i>C1:</i> <code>recommend(user, LO) ← educationally_appropriate(user, LO) ∧ generally_appropriate(LO)</code>	
<hr/>	
<i>C1.1:</i> <code>educationally_appropriate(user, LO) ← appropriate_resource(user, LO) ∧ appropriate_interactivity(user, LO)</code>	
<hr/>	
<i>C1.1.1:</i> <code>appropriate_resource(user, LO) ← user_type(user, type) ∧ resource_type(LO, type)</code>	
Explanation: 'This LO may interest you, since it is a [RESOURCE_TYPE], which is suitable for your [LEARNING_STYLE] learning style'	
Responses:	
+RC1.1.1: 'Accept'	
~RC1.1.1: 'Not sure. Show me more reasons'	
-RC1.1.1: 'Reject. I prefer LOs of the type [TYPE]'	
<hr/>	
<i>C1.1.2:</i> <code>appropriate_interactivity(user, LO) ← user_type(user, type) ∧ interactivity_type(LO, type)</code>	
Explanation: 'This LO may interest you, since it requires [INTERACTIVITY_TYPE] interaction, which is suitable for your [LEARNING_STYLE] learning style'	
Responses:	
+RC1.1.2: 'Accept'	
~RC1.1.2: 'Not sure. Show me more reasons'	
-RC1.1.2: 'Reject. I prefer LOs that require [INTERACTIVITY_TYPE] interaction'	

Table 2. Cont.

<p>C1.2: $generally_appropriate(LO) \leftarrow structure(LO, atomic) \wedge state(LO, final)$ Explanation: ‘This LO may interest you, since it is self-contained’ Responses: +RC1.2: ‘Accept’ ~RC1.2: ‘Not sure. I do not care that the LO is not self-contained’ –RC1.2: ‘Reject. I don’t like it’</p>
<p>C2: $recommend(user, LO) \leftarrow educationally_appropriate(user, LO) \wedge generally_appropriate(LO) \wedge technically_appropriate(user, LO)$</p>
<p>C2.1: $technically_appropriate(user, LO) \leftarrow appropriate_language(user, LO) \wedge appropriate_format(LO)$</p>
<p>C2.1.1: $appropriate_language(user, LO) \leftarrow language_preference(user, language) \wedge object_language(LO, language)$ Explanation: ‘This LO may interest you, since it suits your language preferences: [OBJECT_LANGUAGE]’ Responses: +RC2.1.1: ‘Accept’ ~RC2.1.1: ‘Not sure. Show me more reasons’ –RC2.1.1: ‘Reject. I prefer LOs in [LANGUAGE]’</p>
<p>C2.1.2: $appropriate_format(LO) \leftarrow format_preference(user, format) \wedge object_format(LO, format)$ Explanation: ‘This LO may interest you, since it suits your format preferences: [OBJECT_FORMAT]’ Responses: +RC2.1.2: ‘Accept’ ~RC2.1.2: ‘Not sure. Show me more reasons’ –RC2.1.2: ‘Reject. I prefer LOs with format [OBJECT_FORMAT]’</p>
<p>C3: $recommend(user, LO) \leftarrow educationally_appropriate(user, LO) \wedge generally_appropriate(LO) \wedge updated(LO)$</p>
<p>C3.1: $updated(LO) \leftarrow date(LO, date) < 5\ years$ Explanation: ‘This LO may interest you, since it is updated’ Responses: +RC3.1: ‘Accept’ ~RC3.1: ‘Not sure. I do not care that the LO is not updated’ –RC3.1: ‘Reject. I don’t like it’</p>
<p>C4: $recommend(user, LO) \leftarrow educationally_appropriate(user, LO) \wedge generally_appropriate(LO) \wedge learning_time_appropriate(LO)$</p>
<p>C4.1: $learning_time_appropriate(LO) \leftarrow hours(LO) < \gamma$ Explanation: ‘This LO may interest you, since it suits your learning time preferences (less than $[\gamma]$ hours to use it)’ Responses: +RC4.1: ‘Accept’ ~RC4.1: ‘Not sure. I do not care about the learning time required to use it’ –RC4.1: ‘Reject. I don’t like it’</p>
<p>Collaborative Rules</p>
<p>O1: $recommend(user, LO) \leftarrow similarity(user, user_k) > \alpha \wedge vote(user_k, LO) \geq 4$ Explanation: ‘This LO may interest you, since it likes to users like you’ Responses: +RO1: ‘Accept’ ~RO1: ‘Not sure. Show me more reasons’ –RO1: ‘Reject. I don’t like it’</p>
<p>Knowledge-based Rules</p>
<p>K1: $recommend(user, LO) \leftarrow similarity(LO, LO_j) > \beta \wedge vote(user, LO_j) \geq 4$ Explanation: ‘This LO may interest you, since it is similar to another LO that you liked ($[LO_j]$)’ Responses: +RO1: ‘Accept’ ~RO1: ‘Not sure. Show me more reasons’ –RO1: ‘Reject. I don’t like it’</p>

For instance, in Table 1, lines 8 to 13 represent *content-based rules* that make recommendations for the student based on its profile. Then, an example of a rule that shows how the recommendation system chooses the learning object LO_1 as the most suitable for *alice* is $appropriate_interactivity(alice, LO_1) \leftarrow user_type(alice, visual) \wedge interactivity_type(LO_1, low)$. This is due to the fact that *alice* has a visual learning style and LO_1 has been labeled as a learning object with low interactivity, which fits with a learning style of type visual.

In addition, line 14 represents a *collaborative rule* that takes information from the students' profile to calculate a similarity degree among them and recommends an LO that was suitable for a similar student *paul*.

Finally, line 15 represents a *knowledge-based rule* that uses information about other LO which the user has already evaluated previously to make a recommendation of a similar LO: LO_2 . Note that, although the fact $vote(alice, LO) \geq 4$ does not appear in the example logic program, in the proposed defeasible argumentation, formalism is not assumed negation as failure. We follow the approach of extended logic programming for non-monotonic reasoning, as proposed in [38], by allowing, in addition to negation as failure, a second, explicit form of negation, written as \sim . Therefore, the non-existence of a fact does not prevent the triggering of those rules that include it and, hence, the knowledge-based rule of line 5 can be used to generate an argument that supports the recommendation of LO_2 . Afterwards, rules can be defeated by the inclusion of new knowledge in the system.

The program that represents the logic of the C-ERS can be queried to resolve if an argument that supports a specific recommendation can be derived. Thus, when a recommendation of LOs for a certain student is submitted to C-ERS, it attempts to derive every $recommend(user, LO)$ defeasible rule by backward chaining facts and defeasible rules and applying a mechanism similar to the SLD (*Selective Linear Definite*) derivation of standard logic programming. Next, it is presented how arguments are defined in this framework, since C-ERS has the ability to produce an argument that supports the literal that can be derived from each defeasible rule:

Definition 1 (Argument). *An argument \mathcal{A} for h ($\langle h, \mathcal{A} \rangle$) is a minimal non-contradictory set of facts and defeasible rules that can be chained to derive the literal (or conclusion) h .*

For instance, it can be derived from the program represented in Table 1 two arguments that support the recommendation of LO_1 to *alice* ($Arg_1 = \langle recommend(alice, LO_1), \{educationally_appropriate(alice, LO_k), generally_appropriate(LO_k)\} \rangle$ and $Arg_2 = \langle recommend(alice, LO_1), \{similarity(alice, paul) > \alpha, vote(paul, LO_1) \geq 4\} \rangle$). Moreover, other arguments to support the suitability of LO_1 for *alice* can be derived. Some examples of such arguments would be: $educationally_appropriate(alice, LO_1)$, $\{appropriate_resource(alice, LO_1), appropriate_interactivity(alice, LO_1)\}$. In addition, it can be also derived one argument that supports the recommendation of LO_2 to the student *alice* ($Arg_3 = \langle recommend(alice, LO_2), \{similarity(LO_2, LO) > \beta, vote(alice, LO) \geq 4\} \rangle$).

In our argumentation formalism, an argument can receive an *attack* from other arguments that *rebut* them (for instance by proposing the opposite conclusion) or *undercut* them (i.e., by attacking clauses of their body). Following the example above Arg_1 and Arg_2 attack Arg_3 while Arg_3 attacks Arg_1 and Arg_2 .

Definition 2 (Attack). *An argument $\langle q, \mathcal{B} \rangle$ attacks argument $\langle h, \mathcal{A} \rangle$ if we can derive $\sim h$ from \mathcal{B} or if q implies that one of the clauses of \mathcal{A} does no longer hold (there is another argument $\langle h_1, \mathcal{A}_1 \rangle$ used to derive $\langle h, \mathcal{A} \rangle$ such that $\Pi \cup \{h_1, q\}$ is contradictory).*

Attacks between arguments are addressed through the use of a measure of probability that calculates the likelihood of an argument succeeding based on the aggregated probability of the facts and clauses in the body of the rules employed to build the argument. Therefore, C-ERS uses a simplified *probabilistic argumentation* framework that maps values of probability to the arguments and then aggregates such values to calculate a *suitability* score for classifying and recommending LOs [39].

Definition 3 (Argumentation Framework). *An argumentation framework is defined in C-ERS as the tuple (Arg, P_{Arg}, D) where Arg is a set of arguments, $D \subseteq Arg \times Arg$ is a defeat relation, and $P_{Arg} : \rightarrow [0 : 1]$ represents the probability that an argument holds.*

To calculate the probability of an argument Arg (where $Arg = \langle h, \mathcal{A} \rangle$):

$$P_{Arg} = \begin{cases} 1 & \text{if } \mathcal{A} \subseteq \Pi \\ \frac{\sum_{i=1}^k P_{Q_i}}{k} & \text{if } \mathcal{A} \subseteq \Delta \mid h \leftarrow Q_1, \dots, Q_k \end{cases} \quad (1)$$

A fact has probability of 1. You can compute the probability of a defeasible rule as the mean of the probabilities for the literals Q_1, \dots, Q_k that form their body. For instance:

- $1 \leftarrow$ if they are facts.
- $0 \leftarrow$ if it is not possible to be solved.
- $P_{Q_i} \leftarrow$ if they are derived from other defeasible rules.

The *suitability* value of a recommendation is then calculated as the product of the probabilities of all its supporting arguments.

Definition 4 (Defeat). An argument defined as $\langle q, \mathcal{B} \rangle$ defeats another argument $\langle h, \mathcal{A} \rangle$ in C-ERS if \mathcal{B} attacks \mathcal{A} and $P_{\mathcal{B}} > P_{\mathcal{A}}$.

For instance, Arg_1 and Arg_2 would have probability 1 and Arg_3 probability 0.5 (since we have no information on the vote that *alice* assigned to *LO* in the past and the fact $\text{vote}(\text{alice}, LO) \geq 4$ cannot be resolved). Therefore, following the above defeat relation, Arg_1 and Arg_2 would defeat Arg_3 and LO_1 would be recommended to *alice*. Otherwise, if different arguments to support the recommendation of different LOs would still hold at the end of the recommendation process, such LO with the greatest *suitability* value would be recommended.

3.2. Conversational Process

As pointed out in Section 1, an extra benefit from our argumentation-based recommendation technique is the straightforward way to generate explanations from arguments. These explanations are justification texts that can be offered to the user to persuade them to try certain LOs and to explain why the system has proposed a specific LO. With each explanation, the user can interact with the system by selecting one of the three possible pre-defined responses (one to accept the explanation and hence the recommendation, one to ask for more justifications, and another one to reject the recommendation). Thus, the current version of C-ERS implements a constrained mode of user interaction that does not provide a natural language interface.

Table 2 shows an example of the explanations that the system is able to generate. Note that the system do not provide explanations for all arguments that the system is able to generate from rules, but only for those rules that represent ‘leaves’ in the rule tree. These represent the deeper clauses that more general ‘recommendation’ rules aggregate to support recommendations. This simple design decision allows us to reduce the complexity of the dialogue and to improve the system’s expressiveness.

In addition, the natural order to perform the backward chaining of rules and facts to derive arguments that support recommendations allows us to establish a conversational process between the C-ERS and the user. By this process, the system is able to elicit the actual preferences of the user and allows him/her to correct the system’s wrong assumptions. Concretely, the following *priority orderings* have been established for rules and their associated explanations (as a result of the possible combinations among the different types of rules):

- **P1: First, show arguments that suit the student profile and preferences (CONTENT-BASED ARGUMENTS):** C1.1.1 > C1.1.2 > C1.2 > C2.1.1 > C2.1.2 > C3.1 > C4.1 > K1 > O1 > G1 > G2

- **P2: First, show arguments that suit the profile and preferences of similar users (COLLABORATIVE ARGUMENTS):** O1 > C1.1.1 > C1.1.2 > C1.2 > C2.1.1 > C2.1.2 > C3.1 > C4.1 > K1 > G1 > G2
- **P3: First, show arguments that suit the usage history of the users (KNOWLEDGE-BASED ARGUMENTS):** K1 > C1.1.1 > C1.1.2 > C1.2 > C2.1.1 > C2.1.2 > C3.1 > C4.1 > O1 > G1 > G2
- **P4: First, show arguments that justify the format of the object (GENERAL ARGUMENTS):** G1 > G2 > C1.1.1 > C1.1.2 > C1.2 > C2.1.1 > C2.1.2 > C3.1 > C4.1 > K1 > O1

Figure 1 illustrates the sequence of explanations that the system shows, depending on the rules that have been triggered to generate arguments for the recommendations and the decisions made by the user in each step of the dialogue.

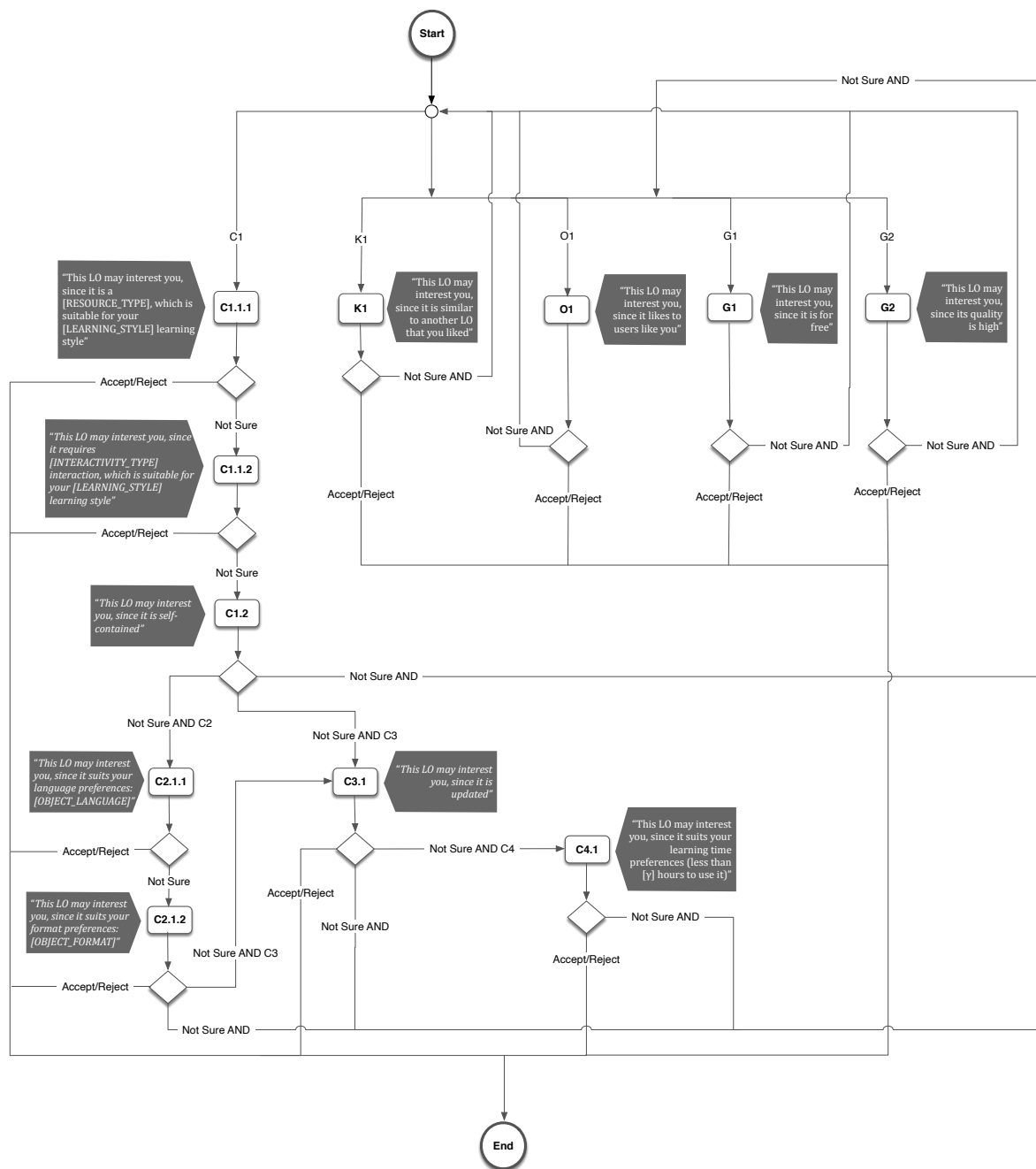


Figure 1. Steps of the conversational dialogue between the C-ERS and the user.

4. Evaluation

To evaluate C-ERS, an *online* evaluation to demonstrate the ability of the system was performed:

1. To provide effective recommendations that suit the students' profile and learning objectives (*effectiveness*).
2. To persuade students to try specific LOs (*persuasiveness*).
3. To elicit the actual preferences of the user by allowing him/her to correct the system's assumptions (*scrutability*).

4.1. Methodology

For the evaluation tests of our C-ERS, a prototype was implemented that uses 75 LOs from FROAC (Federation of Learning Objects Repositories of Colombia). These LOs include learning materials from computer science such as database, systems, algorithms, programming, etc. The LOs are also provided with metadata that represents information about the title, description, keywords, type, language, format, and interactive level. All learning objects were in Spanish and 20% of them also had an English version.

The tests were performed by a group of 50 bachelor degree students in Computer Science of the Universidad Nacional de Colombia. The students signed an informed consent to participate. During the registration, the system requested from these students information about their topics of interest (interest keywords) and level of education (years as bachelor degree student). They also completed a test to determine their appropriate learning style. Thus, the system was able to register the next features for each student:

- Personal data: ID, name, surname, sex, date of birth, nationality, city of residence, address, language, phone, and mail.
- Student's educational preferences:
 - Interactivity level: high human–computer interaction LOs (preferred by nine students), medium human–computer interaction LOs (preferred by 38 students), or LOs that focus on presentation of content (preferred by three students).
 - Preferred language: all students mother tongue was Spanish, and thus all preferred LOs in Spanish.
 - Preferred format: *jpeg* was selected by nine students, *mp4* by eight students, *pdf* by 31 students, and 'other formats' by two students.
- Learning Style: to model the learning style of each student, it was followed the VARK (<http://vark-learn.com/>) model. The model classified 25 as visual students, 6 as auditory, 12 as reader, and 7 as kinesthetic.
- History of uses: for each LO ranked by the student, the system stores its ID, the rating assigned, and the date of use.

After the registration, the evaluation tests were performed as follows. Each student participated in 10 recommendation processes, where the system recommended LOs that match his/her profile for a specific search (a list of specific keywords). As pointed out in Section 3.2, the system could follow four different argumentation strategies, which result in different orderings to show arguments (see Figure 1 for details on which arguments succeed to others). In the recommendation processes, the system was settled to follow the following priority orderings:

- Recommendation processes 1–2: P1 (show content-based arguments first)
- Recommendation processes 3–4, P2 (show collaborative arguments first)
- Recommendation processes 5–6, P3 (show knowledge-based arguments first)
- Recommendation processes 7–8, P2 (show general arguments first)

- Recommendation processes 9–10, random

Among these recommendation processes, the system computed the best top five recommendations (those with the largest number of generated arguments) and provided the student with the best of the list (the one with the more arguments generated). Then, the student had to provide an initial rating for the LO recommended.

Furthermore, for 5 out of these 10 recommendation processes (even recommendation processes, i.e., 2, 4, 6, 8, 10), the system showed an explanation for the LO recommended. When no explanation was provided (odd recommendation processes), the system's interface allowed the student to 'Accept' the recommendation, 'Reject' it, or 'Ask for a justification'. When an explanation was provided, the student could 'Accept' the LO, 'Reject' the LO, or else, 'Reject' the explanation (and hence, the underlying argument).

When the student *accepted or rejected the recommendation* at the first step of the recommendation process, he/she was able to run the next recommendation process of the evaluation tests. When the student *rejected the argument (and its underlying explanation)*, different situations could arise. On the one hand, the explanation could be rejected just because it was not sufficiently convincing for the student and he/she asked for further explanations (e.g., 'Not sure. Show me more reasons' or 'Not sure. I do not care that the LO is not updated' in Figure 1). In this case, the next available argument (if any) was used to generate a new explanation in a second step of the recommendation process. If the student rejected the explanation again, a new one was provided in a new step of the recommendation process and so on until he/she accepted or rejected the LO or there were no longer available arguments. In that case, the recommendation was just rejected.

On the other hand, the explanation could be rejected due to the content of the explanation itself, which may include errors or outdated information in the system's assumptions about the student's preferences (e.g., 'Reject. I prefer LOs of the type [RESOURCE_TYPE]' or 'Reject. I prefer LOs in [SPECIFIC_LANGUAGE]'). This entails a change in the profile of the student and the system had to compute again the recommendations taking this new information into account (which yielded to the next recommendation process, if the student had not already completed the 10 recommendation processes).

Finally, at the end of each recommendation process (after all possible steps), the student was urged to revise his/her initial rating and change it if desired. With these evaluation tests, a database of 472 ratings was obtained from the total number of 500 recommendation processes (50 students × 10 recommendation processes each). An schema of this process is shown in Figure 2.

4.2. Results

The experiments evaluated the *effectiveness* of C-ERS to provide recommendations that suit the students' profile and learning objectives. The advantages of the argumentation-based recommendation over other approaches in this domain, such as content-based, collaborative filtering, and knowledge-based, was previously demonstrated in our previous work [4]. Therefore, we do not intend to evaluate the whole learning process of students, as proposed, for instance, in Kirkpatrick's model [40], which tries to evaluate the learning process from reaction, to learning, to behavior, and to organisational performance. Here, we just focus on Kirkpatrick's level 1, evaluating the reaction of the students to the learning objects proposed by the system. As shown in Figure 3, for all recommendation processes, we computed the average of original ratings that students provided to the learning objects that the system recommended to them in two separated groups, the average rating of the learning objects that were provided in recommendation processes where explanations were shown ('With Argument' in the figure), and the average rating to those learning objects that were provided in recommendation processes where the system did not show any explanation (just the object, 'Without Argument' in the figure). In addition, 25th and 75th percentiles are marked by the boundaries of the boxes, and the average ratings by dots.

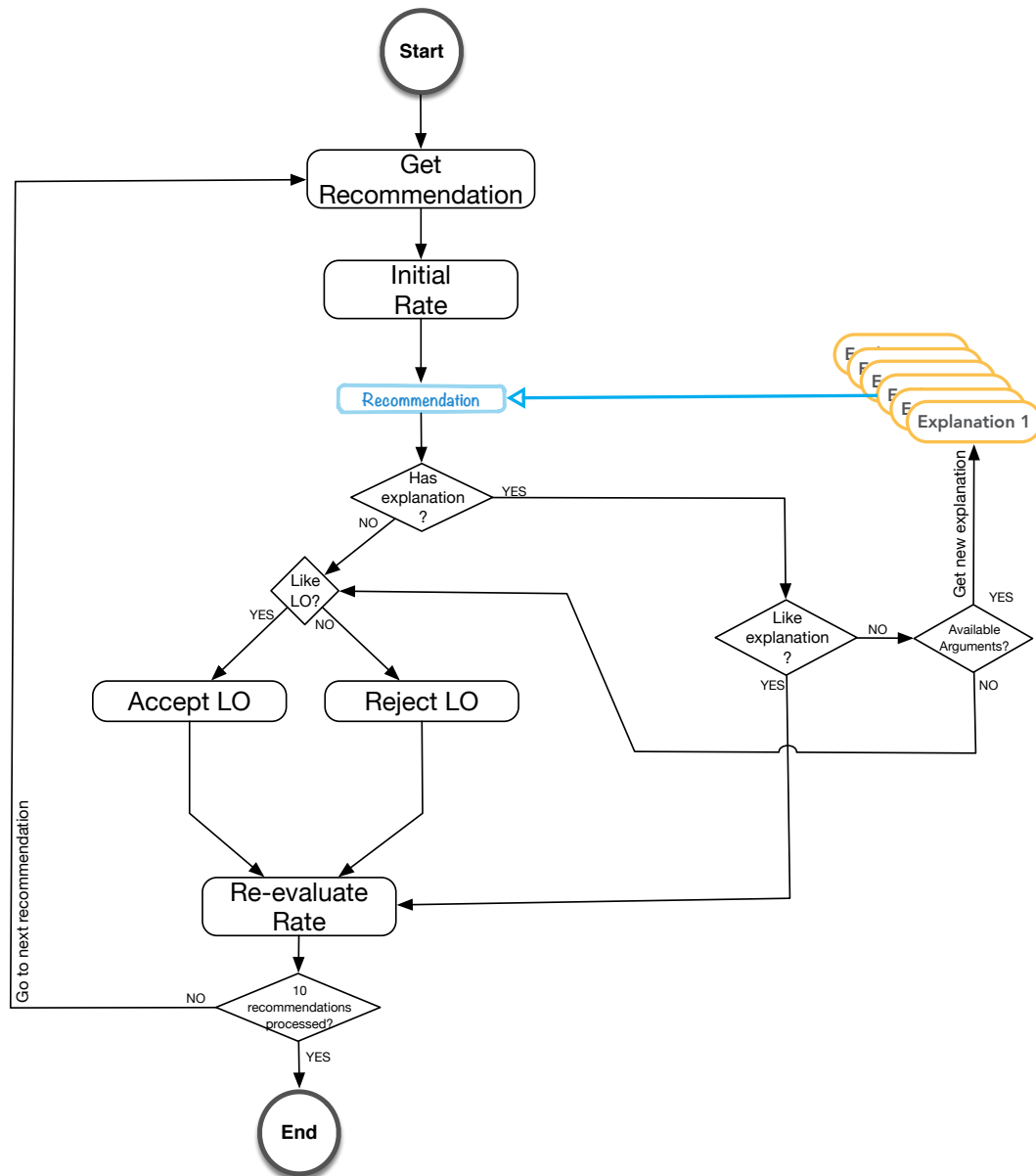


Figure 2. Schema followed by the described experiment.

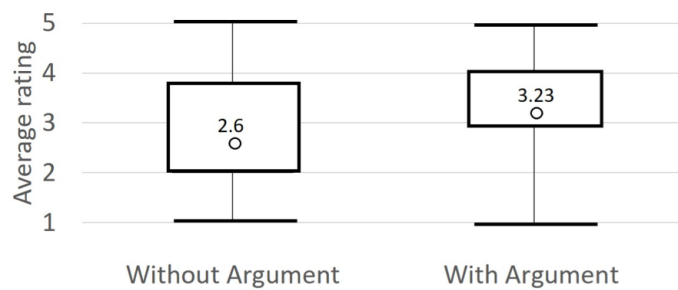


Figure 3. Average ratings provided by the students without and with arguments and their associated explanations.

Moreover, Figure 4 shows exactly how often each rating was chosen. As we can see in the figures, students provided higher ratings to those LOs that presented explanations, which demonstrates the

advantages of using explanations to offer effective recommendations. Furthermore, the number of highly rated objects (3 to 4) increased when they came with explanations.

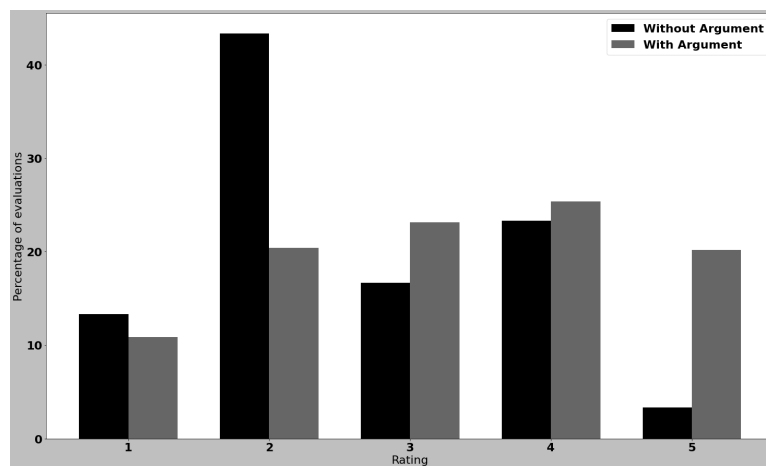


Figure 4. Histogram with the rating comparison between students without and with arguments.

We consider as a baseline the case in which students do not receive explanations attached to the recommendation (without argument), since the hypothesis to be demonstrated is that the ratings assigned to ‘justified’ recommendations are higher, in other words that students are more satisfied when they understand the reason why the system provides them with a specific recommendation. On the data shown in Figure 3, the tests of statistical significance show a *p*-value of 0.0046 and *g* of Hedges of 1.27 (effect size significant for samples with different variances –1.21 without arguments; 1.64 with arguments), which demonstrate the significance of the results obtained.

In Figures 5 and 6, experiments also evaluated the average ratings provided by the students for each priority ordering to show the explanations, and how often each rating was chosen for each priority ordering. As depicted in the figures, all priority orderings seem to have a similar performance, with a slight increase in the average rating and in the quantity of LOs that got better ratings for P3 (first show arguments that suit the usage history of the users).

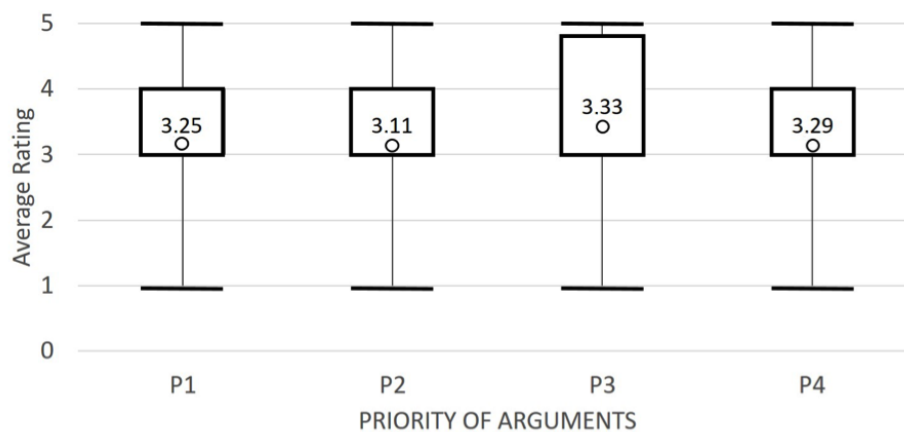


Figure 5. Average ratings provided by the students for each priority ordering to show the explanations.

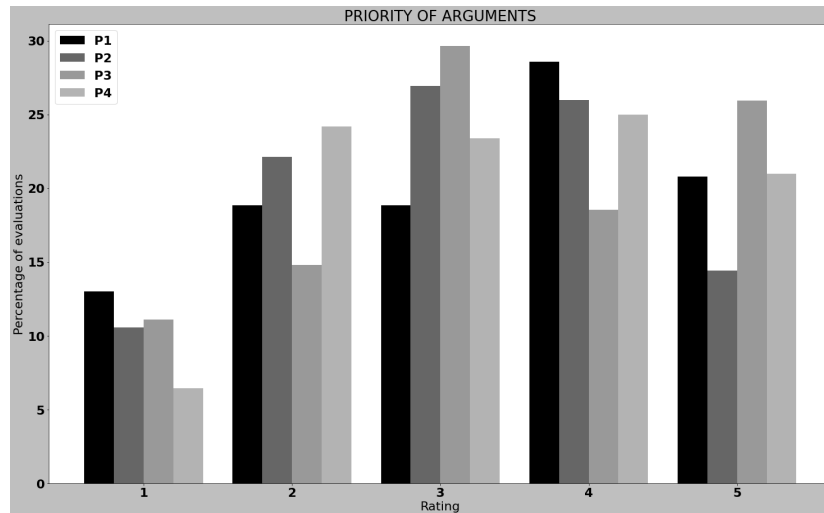


Figure 6. Histogram with the rating comparison between students for each priority ordering to show the explanations.

To evaluate the *persuasive power* of C-ERS, the initial rating provided by the student to LOs and the final one after the ‘conversation’ between the system and the student were compared. Figure 7 shows the average percentage of decreased, improved, and unchanged ratings throughout the conversational recommendation process (from the 1st iteration to the end). As illustrated in the figure, a wide percentage of ratings were improved by the interchange of explanations between the C-ERS and the student, which demonstrates the persuasive power of explanations to improve the opinion of students about the LOs recommended.

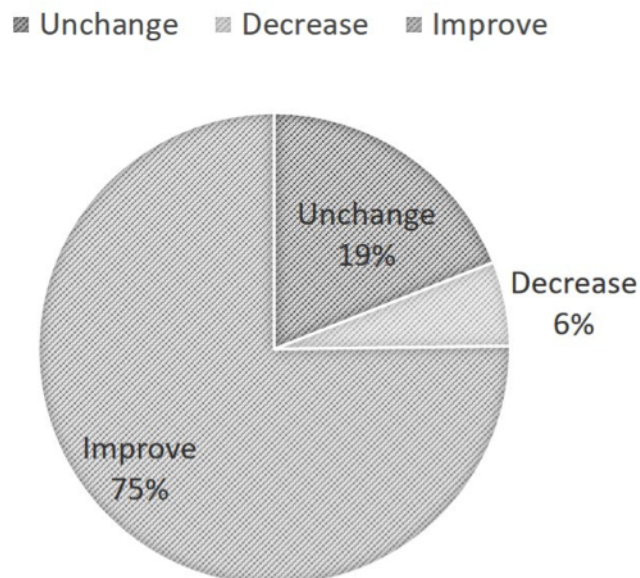


Figure 7. % of decreased, improved, and unchanged ratings in the conversation (from the 1st iteration to the end).

In addition, Figure 8 shows the average percentage of students that accepted the recommendation in the 1st iteration (without and with explanations), and at the end of the dialogue with the system. As illustrated in the figure, most students preferred to receive explanations before they accept the LO

recommended, which again demonstrates the effectiveness of explanations to influence the students' will to accept recommendations.

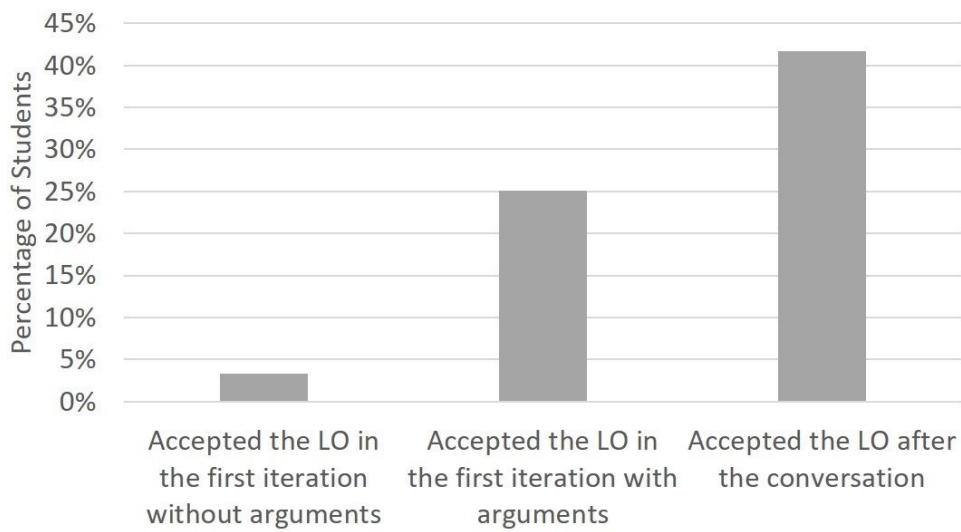


Figure 8. Average % of students that accepted the recommendation in the 1st iteration (without and with explanations), and at the end of the conversation.

To evaluate the *scrutability* of C-ERS, the average percentage of students was analyzed that rejected explanations because they changed any of their initially declared preferences. Results shown in Figure 9 demonstrate that a significant percentage of students decided to change their preferences at any step of the recommendation process (especially those regarded to the interactivity type required to use LOs). On average, over the total number of the recommendation processes where they participated, 38% of students decided to change any preference. Our C-ERS system is able to capture these changes and allow students to indicate that it has made a wrong inference about his/her preferences, which is crucial in this educational recommendation domain to provide them with recommendations that actually suit their learning objectives.

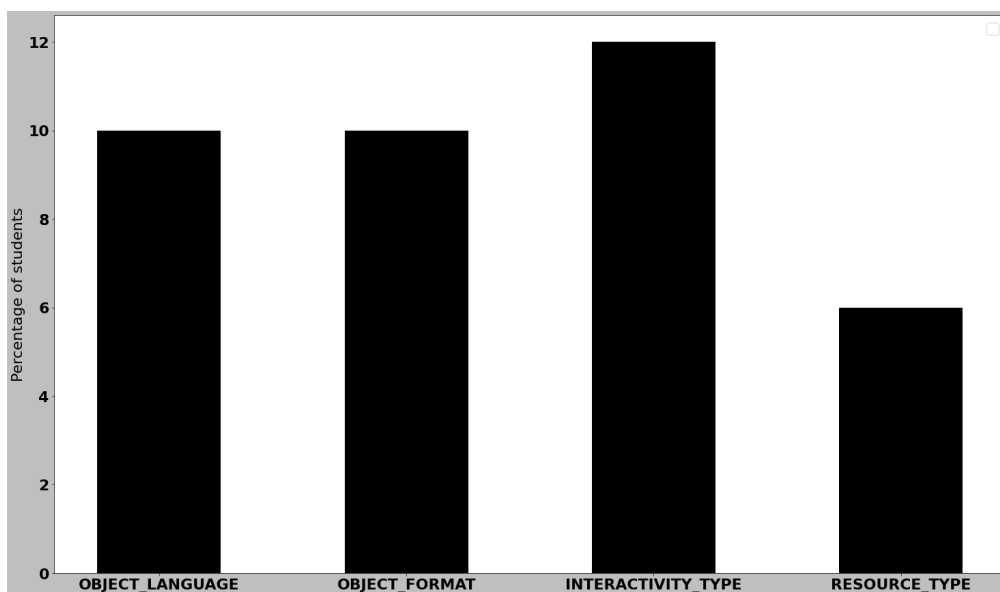


Figure 9. Average % of students that changed their declared preferences.

Moreover, the system performance as the average number of explanations that it has to provide until the user accepts the recommendation was also measured. We got an average number of 2.2 explanations needed to convince the students to accept LOs. This is short enough to ensure that the average time required to end each conversational dialogue is not as excessive as to bore students and make them abandon the recommendation process.

5. Conclusions

An argumentation-based Conversational Educational Recommender System (C-ERS) has been presented in this paper. The proposed system helps students to find the more suitable learning resources considering different aspects such as learning objectives, educational level, and learning style. According to this, the main contribution of the proposed system is that it allows the user to interact with the system and accept or reject the recommendations, providing reasons for such behavior. These reasons are used by the system to convince students to accept the recommendations of the system and make use of the recommended learning objects.

The proposed system has been tested and evaluated with a real group of undergraduate students in the Universidad Nacional de Colombia. In the proposed experimentation, each student participated in 10 recommendation processes, where the system recommended learning objects that match the students' profile for a specific search. Additionally, the system showed an explanation for the recommended learning objects. Results have shown the ability of the system to provide effective recommendations that suit the students' profile and learning objectives, to try to persuade students to use certain learning objects, and to elicit their actual preferences by allowing students to correct the system's wrong assumptions.

Actually, the current system constitutes a proof of concept tested with a set of 50 students of computer science. This could entail some bias in the students' profile. As additional work, we will try to extend the evaluation tests to a large number of students with more heterogeneous profiles. Moreover, we want to study to what extent the new user model acquired with the new preferences elicited during the conversation should be updated and stored permanently or just should be considered as an ad-hoc profile for the current recommendation process.

Finally, as future work, we plan to enhance the interaction mode of the system with a new natural language interface, able to conduct dialogues in natural language with the students. In addition, in our C-ERS explanations are textual versions of arguments. In doing so, we are just using one format of explanations, and, in fact, we acknowledge that this type is not the best type for all learners (clearly, it is more suitable for "readers") [41]. Thus, in future work, we will translate our arguments to different formats of explanations.

Author Contributions: Conceptualization, V.J. and N.D.-M.; Formal analysis, P.R. and S.H.; Investigation, J.P., P.R., and S.H.; Methodology, V.J., N.D.-M., J.P., P.R., and S.H.; Project administration, V.J.; Supervision, V.J.; Writing—original draft, J.P., P.R., and S.H.; Writing—review, V.J., J.P., P.R., and S.H. and editing, J.P., S.H., and V.J. All authors have read and agreed to the published version of the manuscript.

Funding: This work was partially supported by MINECO/FEDER RTI2018-095390-B-C31 project of the Spanish government, and by the Generalitat Valenciana (PROMETEO/2018/002) project.

Conflicts of Interest: The authors declare no conflict of interest. The founding sponsors had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript, and in the decision to publish the results.

References

1. Drachler, H.; Verbert, K.; Santos, O.C.; Manouselis, N. Panorama of recommender systems to support learning. In *Recommender Systems Handbook*; Springer: Berlin/Heidelberg, Germany, 2015; pp. 421–451.
2. Tucker, B. The flipped classroom. *Educ. Next*, **2012**, *12*, 82–83.
3. Zapalska, A.; Brozik, D. Learning styles and online education. *Campus-Wide Inf. Syst.* **2006**, *23*, 325–335. [[CrossRef](#)]

4. Rodríguez, P.; Heras, S.; Palanca, J.; Poveda, J.M.; Duque, N.; Julián, V. An educational recommender system based on argumentation theory. *AI Commun.* **2017**, *30*, 19–36. [[CrossRef](#)]
5. Rodríguez, P.A.; Ovalle, D.A.; Duque, N.D. A student-centered hybrid recommender system to provide relevant learning objects from repositories. In Proceedings of the International Conference on Learning and Collaboration Technologies, Los Angeles, CA, USA, 2–7 August 2015; pp. 291–300.
6. Bridge, D.G. Towards Conversational Recommender Systems: A Dialogue Grammar Approach. In Proceedings of the ECCBR Workshops, Aberdeen, UK, 4–7 September 2002; pp. 9–22.
7. Christakopoulou, K.; Radlinski, F.; Hofmann, K. Towards conversational recommender systems. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, 13–17 August 2016; pp. 815–824.
8. Zhao, X.; Zhang, W.; Wang, J. Interactive collaborative filtering. In Proceedings of the 22nd ACM International Conference on Conference on Information & Knowledge Management, San Francisco, CA, USA, 27 October–1 November 2013; pp. 1411–1420.
9. Rubens, N.; Elahi, M.; Sugiyama, M.; Kaplan, D. Active learning in recommender systems. In *Recommender Systems Handbook*; Springer: Berlin/Heidelberg, Germany, 2015; pp. 809–846.
10. Chen, L.; Pu, P. Critiquing-based recommenders: Survey and emerging trends. *User Model. User-Adapt. Interact.* **2012**, *22*, 125–150. [[CrossRef](#)]
11. Felfernig, A.; Friedrich, G.; Jannach, D.; Zanker, M. Constraint-based recommender systems. In *Recommender Systems Handbook*; Springer: Berlin/Heidelberg, Germany, 2015; pp. 161–190.
12. Mahmood, T.; Ricci, F. Improving recommender systems with adaptive conversational strategies. In Proceedings of the 20th ACM Conference on Hypertext and Hypermedia, Orino, Italy, 29 June–1 July 2009; pp. 73–82.
13. He, C.; Parra, D.; Verbert, K. Interactive recommender systems: A survey of the state of the art and future research challenges and opportunities. *Expert Syst. Appl.* **2016**, *56*, 9–27. [[CrossRef](#)]
14. Vig, J.; Sen, S.; Riedl, J. Tagsplanations: Explaining Recommendations Using Tags. In Proceedings of the 14th International Conference on Intelligent User Interfaces, Sanibel Island, FL, USA, 8–11 February 2009; pp. 47–56. [[CrossRef](#)]
15. Symeonidis, P.; Nanopoulos, A.; Manolopoulos, Y. MoviExplain: A Recommender System with Explanations. In Proceedings of the Third ACM Conference on Recommender Systems, New York, NY, USA, 23–25 October 2009; pp. 317–320. [[CrossRef](#)]
16. Tintarev, N.; Masthoff, J. Designing and evaluating explanations for recommender systems. In *Recommender Systems Handbook*; Springer: Berlin/Heidelberg, Germany, 2011; pp. 479–510.
17. Tintarev, N.; Masthoff, J. Explaining recommendations: Design and evaluation. In *Recommender Systems Handbook*; Springer US: New York, NY, USA, 2015; pp. 353–382.
18. Fogg, B. Persuasive technology: using computers to change what we think and do. *Ubiquity* **2002**, *2002*, 5. [[CrossRef](#)]
19. Yoo, K.H.; Gretzel, U.; Zanker, M. Source Factors in Recommender System Credibility Evaluation. In *Recommender Systems Handbook*; Springer: Berlin/Heidelberg, Germany, 2015; pp. 689–714.
20. Benbasat, I.; Wang, W. Trust in and adoption of online recommendation agents. *J. Assoc. Inf. Syst.* **2005**, *6*, 4. [[CrossRef](#)]
21. Sinha, R.; Swearingen, K. The role of transparency in recommender systems. In Proceedings of the Conference on Human Factors in Computing Systems, Minneapolis, MN, USA, 20–25 April 2002; pp. 830–831.
22. Zapata, A.; Menendez, V.; Prieto, M.; Romero, C. A hybrid recommender method for learning objects. *IJCA Proc. Des. Eval. Digit. Content Educ. (DEDCE)* **2011**, *1*, 1–7.
23. Sikka, R.; Dhankhar, A.; Rana, C. A Survey Paper on E-Learning Recommender Systems. *Int. J. Comput. Appl.* **2012**, *47*, 27–30. [[CrossRef](#)]
24. Salehi, M.; Pourzaferani, M.; Razavi, S. Hybrid attribute-based recommender system for learning material using genetic algorithm and a multidimensional information model. *Egypt. Inform. J.* **2013**, *14*, 67–78. [[CrossRef](#)]
25. Dwivedi, P.; Bharadwaj, K. e-Learning recommender system for a group of learners based on the unified learner profile approach. *Expert Syst.* **2015**, *32*, 264–276. [[CrossRef](#)]
26. Tarus, J.K.; Niu, Z.; Mustafa, G. Knowledge-based recommendation: a review of ontology-based recommender systems for e-learning. *Artif. Intell. Rev.* **2018**, *50*, 21–48. [[CrossRef](#)]

27. Walton, D. Argumentation Schemes and Their Application to Argument Mining. *Stud. Crit. Think. Ed. Blair Windsor Stud. Argum.* **2019**, *8*, 177–211.
28. Briguez, C.; Budán, M.; Deagustini, C.; Maguitman, A.; Capobianco, M.; Simari, G. Towards an Argument-based Music Recommender System. *COMMA* **2012**, *245*, 83–90.
29. Briguez, C.; Capobianco, M.; Maguitman, A. A theoretical framework for trust-based news recommender systems and its implementation using defeasible argumentation. *Int. J. Artif. Intell. Tools* **2013**, *22*. [[CrossRef](#)]
30. Recio-García, J.; Quijano, L.; Díaz-Agudo, B. Including social factors in an argumentative model for Group Decision Support Systems. *Decis. Support Syst.* **2013**, *56*, 48–55. [[CrossRef](#)]
31. Chesñevar, C.; Maguitman, A.; González, M. Empowering recommendation technologies through argumentation. In *Argumentation in Artificial Intelligence*; Springer: Berlin/Heidelberg, Germany, 2009; pp. 403–422.
32. Briguez, C.; Budán, M.; Deagustini, C.; Maguitman, A.; Capobianco, M.; Simari, G. Argument-based mixed recommenders and their application to movie suggestion. *Expert Syst. Appl.* **2014**, *41*, 6467–6482. [[CrossRef](#)]
33. Naveed, S.; Donkers, T.; Ziegler, J. Argumentation-Based Explanations in Recommender Systems: Conceptual Framework and Empirical Results. In Proceedings of the Adjunct Publication of the 26th Conference on User Modeling, Adaptation and Personalization, Singapore, 8–11 July 2018; pp. 293–298.
34. Klačnja-Milićević, A.; Ivanović, M.; Nanopoulos, A. Recommender systems in e-learning environments: A survey of the state-of-the-art and possible extensions. *Artif. Intell. Rev.* **2015**, *44*, 571–604. [[CrossRef](#)]
35. Fleming, N. The VARK Questionnaire-Spanish Version. 2014. Available online: <https://vark-learn.com/wp-content/uploads/2014/08/The-VARK-Questionnaire-Spanish.pdf> (accessed on 10 April 2020).
36. Ricci, F.; Rokach, L.; Shapira, B. *Recommender Systems Handbook*; Springer: Berlin/Heidelberg, Germany, 2015.
37. García, A.; Simari, G. Defeasible logic programming: An argumentative approach. *Theory Pract. Log. Program.* **2004**, *4*, 95–138. [[CrossRef](#)]
38. Gelfond, M.; Lifschitz, V. Classical negation in logic programs and disjunctive databases. *New Gener. Comput.* **1991**, *9*, 365–385. [[CrossRef](#)]
39. Li, H.; Oren, N.; Norman, T. Probabilistic argumentation frameworks. In *Theory and Applications of Formal Argumentation*; Springer: Berlin/Heidelberg, Germany, 2012; pp. 1–16.
40. Kirkpatrick, D.; Kirkpatrick, J. *Evaluating Training Programs: The Four Levels*; Berrett-Koehler Publishers: San Francisco, CA, USA, 2006.
41. Snow, R.E. Aptitude-treatment interaction as a framework for research on individual differences in psychotherapy. *J. Consult. Clin. Psychol.* **1991**, *59*, 205–216. [[CrossRef](#)] [[PubMed](#)]



© 2020 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<http://creativecommons.org/licenses/by/4.0/>).