On Using Conversational Frameworks to Support Natural Language Interaction in Intelligent Tutoring Systems

Romina Soledad Albornoz-De Luise, Miguel Arevalillo-Herráez, and David Arnau

***Abstract*—In this paper, we analyse the potential of conversational frameworks to support the adaptation of existing tutoring systems to a natural language form of interaction. We have based our research on a pilot study, in which the open source machine learning framework Rasa has been used to build a conversational agent that interacts with an existing Intelligent Tutoring System (ITS) called Hypergraph Based Problem Solver (HBPS). This agent has been seamlessly integrated into the ITS to replace the previously available button-based user interface and allow the user to interact with HBPS in natural language. Once appropriately trained, the conversational agent was capable of identifying the intention of a given user utterance and extracting the relevant entities related to the message content, with average weighted F1-scores of 0.965 and 0.989 for intents and entities, respectively. Methodological guidelines are provided to generate a realistic training set that enables the creation of the required Natural Language Understanding (NLU) model and also evaluate the resulting system. These guidelines can be easily exported to other ITS and contexts, to provide an enhanced interaction based on natural language processing methods.**

***Index Terms*—Intelligent tutoring systems (ITS), Interactive learning environments (ILE), Conversational agents, Rasa, Natural language understanding (NLU), Natural language processing (NLP).**

I. Introduction

COMPUTERS and the Internet have enabled the development of interactive learning environments that are able to imitate human tutors and provide personalized instruction to students. In this context, there is a growing interest in using natural language as the primary means of interaction between the user and the computer. This is because discourse-based communication is the natural modality of pedagogy [1] and also because of the multiple benefits of conversational systems, that have surpassed effectiveness when compared to control conditions (static content, classroom teaching, etc.) [2]. As examples of Conversational Intelligent Tutoring Systems (C-ITS), we can mention AutoTutor [3] and its descendants, such as Atlas [4] or Watson Tutor [5], among others.

Recent advances in the deep learning field have caused an unprecedented boost in the performance of Natural Language Processing (NLP) applications [6], including typical tasks such as machine translation, opinion mining, text summarization or emotion classification. Such progress has also helped the development of toolkits and frameworks that support the construction of virtual agents that are able to mimic a natural conversation. Some examples are DialogFlow [7], Amazon Lex [8], IBM Watson [9] and Rasa [10], to name a few. In general, these frameworks use pattern matching and NLP methods to analyse the user input, classify it according to the user intent and extract relevant attributes, which are generally known as entities.

Surprisingly, the impact of these rapid advancements in the NLP area has not been as impressive in C-ITS as it has been in other research areas. Most conversational agents in this field are still based on less effective classical methods, and state-of-the-art technologies have rarely been applied to the development of conversational tutors.

In this paper, we use one of the conversational frameworks mentioned above in a specific use case, with the purpose of advancing towards the evaluation of the potential of such frameworks as a way to reduce the existing gap between NLP developments and their practical application in educational contexts. Using Rasa [10] as a representative example, and taking advantage of its availability as an open source implementation, we show that this tool provides the required support to easily and effectively adapt existing tutoring systems to a more convenient form of interaction based on natural language, at the same time we set out some methodological guidelines to help this transformation. The proposed approach has been tested on an existing ITS called Hypergraph Based Problem Solver (HBPS) [11], [12], [13], by replacing the original interaction mechanism based on buttons and menus by another one fundamentally based on the use of natural language; and can be easily generalized and applied to other tutoring systems, both existing and newly created. The performance evaluation clearly shows that the conversational system reaches an acceptable performance with a relatively low amount of training data. When using 25 different intents and using a training set generated from 6 293 sample utterances, average weighted F_1-scores of 0.965 and 0.989 were obtained for intents and entities, respectively.

The presented functional extension is aligned with the

Manuscript received March 7, 2022. This research has been supported by project PGC2018-096463-B-I00, funded by MCIN/AEI/10.13039/501100011033 and FEDER Una manera de hacer Europa; project AICO/2021/019, funded by Valencian Regional Government (Spain); and grant PRE2019-090854, funded by MCIN/AEI/10.13039/501100011033 and "ESF Investing in your future".

R. S. Albornoz-De Luise and M. Arevalillo-Herráez are with the Departament d'Informàtica, Universitat de València, Avda. de la Universitat s/n 46100 Burjassot, Valencia, Spain. E-mail: {romina.albornoz, miguel.arevalilo}@uv.es

D. Arnau is with the Departament de Didàctica de la Matemàtica, Universitat de València, Avda. Tarongers, 4, 46022 Valencia, Spain. E-mail: david.arnau@uv.es

appearance of other ITS that use natural language as the primary means of interaction and follows a current trend, which has motivated a recent survey [2] and extended across teaching and learning systems in several subject areas, including mathematics [14], [15], [16], natural sciences [17], [18], computing [3], [19], languages [20] and transferable skills [21], [22]. However, these existing systems have adopted an ad-hoc approach to conversational design, generally tightly coupled with the domain and/or the knowledge representation used by the C-ITS. On the contrary, our aim has been the development of exportable methods that can be re-used across different subject areas, by combining the use of open-source technology and generalizable procedures.

The rest of the paper is organized as follows. In section II we describe the current state of conversational engines in the context of C-ITS. In section III we explain the typical tasks required in a typical conversational agent design. In section IV we analyse some typical needs related to data collection. In section V we define and explain the evaluation measures, describe a set of guidelines to generate the models, and present the most relevant results. Section VI analyses the relation between performance and training size. In section VII, we focus on the potential generalization of the methods and results presented in the paper, describe the conversational agent integration and discuss the architectural changes that were required in our case study. Finally, we extract the main conclusions and outline some future work in section VIII.

II. Background and State of the Art

Task-oriented chat systems are becoming frequent for businesses. They are currently used in a wide range of common activities and are mostly addressed to improve customer service e.g. product ordering, technical support or information delivery. Most of these services have been developed by using specialized toolkits that provide a machine learning-based NLU component, which can be trained to identify application-specific intents and extract entities from the user's utterances. These toolkits offer similar functionality and operate in a similar way, despite differences in their architecture that affect their performance. In general, they use a connectionist approach based on the use of deep neural networks; and are trained with a large set of labelled realistic examples of utterances supplied by the developer, which the system processes to build a model that can be later used in production [23].

Although the presence of these specialized toolkits is rapidly increasing in industry, their use is still very scarce in educational settings. On the contrary, most conversational systems in this area use more classical approaches to manage the discourse, typically based on statistical or symbolic approaches.

Statistical methods base their processing on a mathematical analysis of the text corpora. Most C-ITS developed to date in this category rely on topic modelling and combine a Bag of Words representation [24] with Latent Semantic Analysis (LSA) to analyze the conceptual similarity of the input utterance to a set of representative training inputs. Autotutor [25] was one of the first C-ITS to use the LSA statistical technique to represent word knowledge and to evaluate the extent to which a student's answer matched an ideal answer. More recently, BRCA Gist [26] has used LSA to interpret user responses in natural language and determine the amount of conceptual and semantic information in the user's response compared to an ideal response identified by the researchers.

Symbolic techniques are based on rules of speech developed by linguistic experts and attempt to determine the meaning of a sentence by combining syntactic and semantic analysis methods. In Beetle II [14], utterances are translated into an interpretable format by an automated reasoning engine that is capable of representing, reconstructing, and validating the user's message at each step. The student messages consist of a mixture of natural language and mathematical expressions, and a representation of their linguistic meaning is built by combining syntactic and semantic analysis techniques, which take place after assigning a symbolic token to each mathematical expression. Also as part of symbolic methods, some C-ITS have adopted a pattern-matching approach. For example, in [20], authors used patterns that consist of a collection of words and a wildcard to match portions of the user's utterance. Similarly, Oscar [27] used a database of scripts containing pattern-based rules that match a stimulus input to a response. More recently, a Cognitive Assistant named LIZA used a specialized parser that evaluated the user input in the context of the answer, looking for typical phrases that occur in this context [28], and LANA-I [29] combined pattern matching with the cosine similarity measure to match the user utterance with one of the patterns stored in a database.

To improve the flow of the conversation, some other C-ITS have also fused the two previous approaches. For example, Aries [21] integrated regular expressions and LSA to evaluate student responses; and Guru [30], combined LSA with concept maps to improve the overall performance of the conversational system. A recently presented system [31] improved the natural language processing offered by Autotutor by focusing on domain-dependent terminology. They generated a new LSA semantic space by using a corpus of documents related to algebra and defining equivalence sets that were used for pre-processing the student input. In this way, they allowed a better evaluation of synonyms for mathematical concepts.

The last available survey on C-ITS [2] reveals an explosion of publications in 2006 and a certain decline after this date, despite the massive advances in NLP technology. In addition, it does not report approaches based on existing conversational frameworks, which we show in this paper they offer the necessary support for the development of both generic and domain-dependent conversational agents.

Although they are still scarce, conversational agents that use such technologies already exist in education, both with an orientation toward administrative support [32], [33] and for teaching purposes. Some examples that use DialogFlow [7] to match the user intention are ScratchThAI [34], for teaching Scratch to students; Fenboturm chatbot [35], used in a 5th-grade science course; and JavaPAL [36], a conversational agent that complements a Massive open online course on Java programming. An ITS created for supporting procedural training in a 3D virtual environment has also been recently presented in [37], this time using IBM Watson [9] as the

conversational framework. Some of these works evaluate the success or failure of a conversational agent [35] as a technological tool [32], as an educational tool or both [37]. However, they are all presented as ad-hoc solutions tightly coupled to their domain of application.

The absence of standard development methodologies, the slow progress in this area and the scarce use of conversational frameworks in the C-ITS field have motivated this experimental work, which aims to analyze the potential of existing frameworks to support the construction of conversational systems in interactive learning environments. In this study, we have used a widely accepted conversational framework to adapt the button-based original user interface of HBPS. This system focuses entirely on the translation stage of the algebraic/arithmetic word problem-solving process and adopts an unguided approach that does not impose any restrictions on the solution path. HBPS is able to check the validity of user entries, provide aids that are consistent with the student's reasoning and simultaneously monitor multiple alternative solution paths. However, the interface was based on the use of radio buttons that allowed learners to manually select their intent (assign a letter to denote a quantity, define a quantity by using an equation or set an equation); and calculator-like components that let them introduce expressions and equations by using previously available quantities. For the purpose of this research, the existing interface has been replaced by a conversational one that allows users to fully interact with the system in natural language, by using a chat system whose operation is supported by the open source Rasa framework.

III. Conversational Agent Design

In a C-ITS, the conversational agent needs to be able to precisely understand a user's input to trigger the most adequate educational response. Conversational frameworks can be trained to process the user's inputs to determine the *intent*, which represents the idea or goal behind each message; and extract meaningful pieces of structured information associated with the intent that are called *entities*. In our case, if the user writes "I would like to define x as the number of rabbits", the system should be able to determine that the intention is to assign a letter to a quantity and also identify "x" and "number of rabbits" as two entities associated with the intent, which refer to the letter and the quantity name, respectively. The abilities of conversational frameworks go beyond this interpretation function and also cover the generation of an appropriate utterance as an answer, as well as the control of the dialogue. However, ITS usually incorporate alternative methods to provide this function and hence we focus on the Natural Language Understanding (NLU) part of the conversational framework.

Detecting the intent from a user message is a well-known machine learning problem and it is done by using text classification methods. To this end, the textual input first needs to be converted into a numerical format that can be processed by the classifier, both in the training and inference phases. This conversion can adopt multiple forms, but it generally requires the application of tokenization and featurization methods. Tokenization splits the text into smaller lexical units, and featurization transforms the resulting lexical units into meaningful numbers. At training time, a classifier learns from the features derived from a set of labelled samples in raw text format. At inference time, the same tokenization and featurization algorithms are used to transform the raw data into the numeric format, and the trained model is used to make predictions.

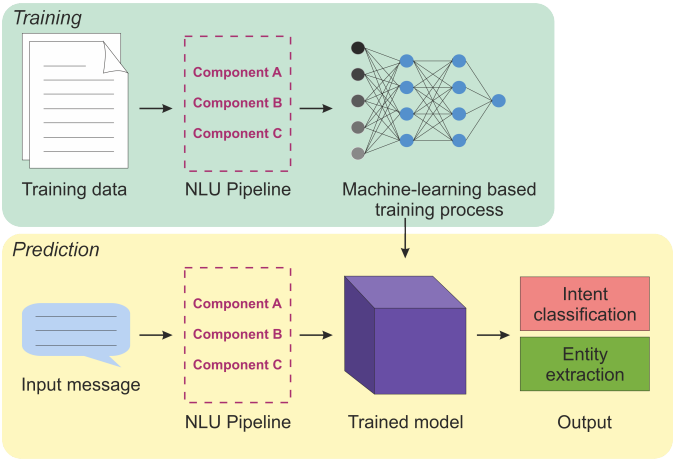

Fig. 1. Training and prediction phases of the machine learning model. The training data gets into the machine learning model to create the training model. The input message is fed to the trained model, that will perform intent classification and entity extraction.

Rasa provides a fully configurable and modular architecture that allows the designer to customize its behaviour by using a series of ready-to-use standard components for a variety of tasks, including pre-processing, intent classification, entity extraction and response selection. The framework also allows the designer to program and add custom NLU components, to perform other tasks not supported by default, such as spell-checks or sentiment analysis. The selected components are specified in a configuration file that defines the so-called NLU pipeline. This pipeline is composed of a series of components that are executed sequentially in the same order as they are stated, taking into account that each one of them may use the outputs of any preceding element, or state specific requirements that need to be satisfied. Fig. 1 illustrates how this pipeline is used in the training and prediction stages. For the final classification, Rasa implements the recent Dual Intent and Entity Transformer (DIET) neural network architecture [38], which is generally used by default. DIET provides the ability to simultaneously handle intent classification and entity recognition by using the outputs of the plug-and-play featurizer components specified in the NLU pipeline, which may include state-of-the-art pre-trained embeddings such as BERT [39], GloVe [40] or ConveRT [41].

IV. Data Gathering

The performance of a text classification system depends on many factors, such as the amount and quality of the training data, the specific techniques used for tokenization, featurization and classification, and how well they all fit together as a single system. In this section, we describe our methodology to construct adequate training and test sets, and

also to optimize the components selection to maximize the network performance.

A. Construction of the Train Corpus

To perform language understanding, the classifier first needs to be trained. This training is done by using a dataset, which ideally should be composed of a sufficiently large collection of realistic text samples. As the conversational system is trained by using the information in this dataset, its completeness is a fundamental factor that largely affects the performance of the final system. Hence, it should include representative samples of every intent, along with the entities associated with each sample.

To be able to build a comprehensive dataset for training the model, a first data collection experience was carried out, with the participation of real teachers and students. In this experience, the ITS interface was kept, but the computer system was replaced by human tutors who personally interacted with the students in a one-to-one setting. The objective of this experience was to record real conversations between tutors and students, so as to collect a sufficiently large database containing different ways of expressing the potential intentions that can arise in a conversation.

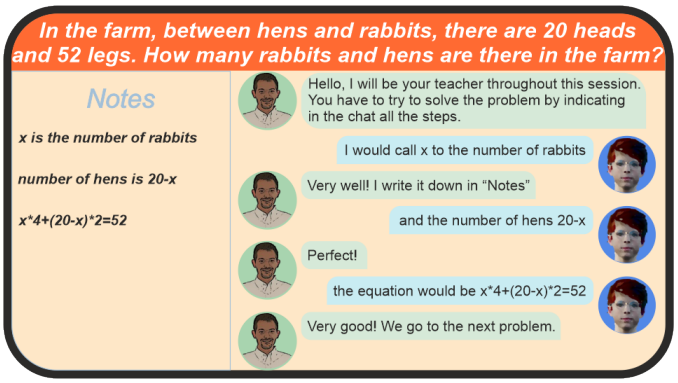

Fig. 2. Fragment of a translated conversation between a teacher and a student.

The group of students consisted of 37 secondary school students (14-15 years old), and the group of tutors was made up of 2 experienced teachers and 35 undergraduate students coursing a Master's Degree in Secondary Education Teacher Training in Spain, who also assumed a teacher role. Each student had to solve a series of 3 verbal algebraic problems, interacting with the teacher through the chat system. The teacher used the same chat to answer questions or provide adequate feedback to the student. Fig. 2 shows a fragment of a real conversation between a teacher and a student.

All utterances recorded in this experience went through a data-cleaning process, in order to eliminate duplicates or entries that clearly escaped the application domain, correct typographical mistakes, and finish incomplete sentences. Despite that most NLP datasets do not undergo such fixing and correction to respect the original wild nature of the sentences, this step was required in our case to be able to seamlessly apply the data augmentation process described in Section IV-B. After cleaning, we had a total of 4 400 messages. 1 756 of them were messages from students, and the remaining 2 644 utterances were from the teachers. Next, the dataset was analysed to identify all different potential intents and entities. A total of 25 different user intents and 5 entities were identified. Intents are listed in Table I, along with a brief explanation for each type and some representative examples that illustrate their meaning. The 5 entities are shown as columns in Table II, which relates them to the intents in which they may be present. Utterances associated with intents not included in this table do not need to carry any additional information other than that implicitly contained in the intent.

Then, we tagged the entities that appeared in each message, if any. All messages were originally in Spanish but were manually translated by the research team into English language, and revised by a native speaker. The objective of this translation was to produce a C-ITS that could be finally used in English language, in order to increase the number of potential users. To complete the training set, some members of the research team added invented representative examples for each intent to the available student entries, generating a corpus $\mathfrak{C}_1$ composed of a total of 6 293 sample utterances. We then corrected typing mistakes to ensure that all words were available in a standard dictionary. The resulting database was then codified in the format accepted by Rasa, in which the intent and entities for each utterance are indicated by using a specific notation. Fig. 3 illustrates this format by using some real samples from the dataset. The *Define_letter* intent has two custom entity labels: *variable* and *description*, while the *Help* intent does not have any labelled entities associated with it.

We shall remark that the definition of intent and entity labels is generally a manual task that must be taken with care. As a rule of thumb, utterances that trigger the same functional response should be grouped together under the same intent, and this grouping requires a meticulous analysis of the available utterances against the expected system responses. In our particular case, for example, note that we differentiate between the intents "define letter" and "define letter missing description", but such a distinction is not made for the intent "expression". In the first case, the system response is different. A description is required so that the system can identify the quantity the user is referring to. Thus, if the description is missing, the system needs to explicitly ask the user to provide a description. However, when the user introduces an expression without a description, the system is able to automatically induce the description from its knowledge base, and then triggers exactly the same response as when a description is explicitly given along with the expression. These considerations are indeed application-dependent, and the definition of the sets of intents and associated entities is generally a complex task with a dominant hand-crafted component that underpins the design of conversational systems.

B. Data Augmentation

When training data is limited, data augmentation methods help increase the number of training samples. This is usually done by generating synthetic data with a known label or adding slightly modified copies of the existing samples. In computer

TABLE I
DESCRIPTION OF IDENTIFIED INTENTS. TO EASE READING, THEY HAVE BEEN GROUPED INTO CATEGORIES. A DESCRIPTION FOR EACH INTENT IS PROVIDED IN THE MIDDLE COLUMN, AND EXAMPLES ARE GIVEN IN THE RIGHT-SIDE COLUMN

Intent Label	Description	Example user utterances
Student answer The user responds to a system question		
affirm	Confirmation or affirmative utterances	- Yes - Ok
deny	Negative answers	- no - never
Info request The user asks a question or requests help		
bot challenge	Questions to challenge the bot	- Are you a robot? - Are you a teacher?
change language	The user wants to change language	- I would like to change the language - I want to write in Spanish
define equation question	Questions about equations	- I don't know how to define a new equation - How do I input an equation?
define letter question	Questions about letters	- can I define a new letter? - what is a variable?
define numeric value question	Questions about numeric quantities	- new number - I want to define a new number
get letter meaning	The user asks for the meaning of a letter	- Can you remind me what h means? - remember me x
help	The user asks for help	- I honestly don't know how to continue - Today, I think I am not able to solve this knotty problem
Solving steps The user's utterance is related to a problem solution step		
define letter	The user enters a letter and a description	- c: is animals in the first farm - t= total animals
define letter missing description	The user attempts to define a quantity by using a letter, but does not provide a description	- I want to define a variable x - y new unknown variable
equation	The user enters an equation	- x*4+(20-x)*2=52 - 45x+6.2y=5.75
expression	The user enters an expression (and possibly a description)	- Ok, then 20x+52y equals the number of animals, right? - in the second farm we have 2+6 animals
number	The user enters a number (and possibly a description)	- there are 13 rabbits - total number of animals in a farm is 20
quantity description	A description for a quantity	- number of legs per rabbit - number of hens
Solving step incorrectly formulated The user attempts to progress in the solution, but the format does not follow a mathematical notation		
equivalence	The user enters an equality without using the "=" sign	- x is 156-y - I would say that c and 2+p take the same value
multiple intents at once	The user tries to do several things at once, e.g. define two different quantities	- There are x chickens and 20-x rabbits - Let's say that x = number of chickens, and y : number of rabbits
word operation	The user enters a mathematical expression by using words	- 87 is the sum of x and y - The farmer's animals are 3 plus 4 plus 2 plus 6
Statement Typical interaction statements not related to the problem solution		
goodbye	The user wants to close the conversation	- bye teacher! - bye
greet	The user greets	- Hello - Hi, how are you
insult	The user enters an insulting message	- Are you drunk, my friend? - Are you stupid or something?
mood great	The user feels good	- I feel very good today - I am happy
thanks	The user thanks the system	- thank you very much for your help - yes, thanks!
wait	The user requests a moment to think	- one moment - I'm thinking about it
out of scope	The user enters an out-of-domain message	- I also have a rabbit in my garden - I don't care much about cows

vision, it is very common to apply some typical operations on the already labelled samples, e.g. horizontal or vertical flips, rotations, scaling, crops, translations or noise additions. In NLP, data augmentation methods attempt to generate sentences that use different words but preserve the context of the original text. The most typical ones are backtranslation [42], which translates a sequence into another language and then translates it back to the original language; and the so-called Easy Data Augmentation (EDA) techniques [43], which consists of a set of random token-level perturbation operations that include random insertion, deletion, swap, and synonym replacement.

A visual inspection of the new samples generated by the backtranslation method supported the consistency of the approach in our case. We used the Google Translate Application

TABLE II
INTENTS WITH TAGGED ENTITIES. ENTITIES THAT CAN POTENTIALLY BE ASSOCIATED WITH EACH INTENT ARE SHOWN, AND MARKED AS OPTIONAL OR COMPULSORY DEPENDING ON WHETHER THEY CAN OR MUST APPEAR, RESPECTIVELY

	Entity				
Intent	description	equation	expression	number	variable
define letter	compulsory				compulsory
define letter missing description					compulsory
equation		compulsory			
equivalence			exactly two entities must appear		
expression	optional		compulsory		
number	optional			compulsory	
quantity description	compulsory				

```
version: "2.0"
nlu:
- intent: Equivalence
  examples: |
    - [9](number) is equal to [3+y](expression)
    - [x](variable) is [20-y](expression)
    - [10y+8x](expression) is [(x+y)9.5](expression)

- intent: Define_letter
  examples: |
    - [c](variable) : [number of chickens](description)
    - [t](variable) = [total animals](description)
    - Let [d](variable) be the [distance](description)

- intent: Expression
  examples: |
    - [490/x](expression)
    - [number of hens](description) is [20-x](expression)
    - [head count](description) is [x+y](expression)

- intent: Equation
  examples: |
    - the equation is [x+y = 20](equation)
    - [x*4+(20-x)*2 = 52](equation)
    - the second equation can be [z(w+2) = 490](expression)

- intent: Number
  examples: |
    - [20](number)
    - there are [13](number) [rabbits](description)
    - [total legs](description) = [52](number)

- intent: Help
  examples: |
    - I honestly don't know how to continue
    - Can you help me solve this problem?
    - This is impossible to solve!
```

Fig. 3. A sample dataset in Rasa format. For each intent, a series of examples are given. In each example, entities are annotated by using squared brackets to denote their content and specifying the entity category in brackets.

Programming Interface (API) on 108 different languages, namely: afrikaans, albanian, amharic, arabic, armenian, azerbaijani, basque, belarusian, bengali, bosnian, bulgarian, catalan, cebuano, chichewa, chinese (simplified), chinese (traditional), corsican, croatian, czech, danish, dutch, esperanto, estonian, filipino, finnish, french, frisian, galician, georgian, german, greek, gujarati, haitian, creole, hausa, hawaiian, hebrew, hindi, hmong, hungarian, icelandic, igbo, indonesian, irish, italian, japanese, javanese, kannada, kazakh, khmer, kinyarwanda, korean, kurdish, kyrgyz, lao, latin, latvian, lithuanian, luxembourgish, macedonian, malagasy, malay, malayalam, maltese, maori, marathi, mongolian, myanmar, nepali, norwegian, odia, pashto, persian, polish, portuguese, punjabi, romanian, russian, samoan, scots gaelic, serbian, sesotho, shona, sindhi, sinhala, slovak, slovenian, somali, spanish, sundanese, swahili, swedish, tajik, tamil, tatar, telugu, thai, turkish, turkmen, ukrainian, urdu, uyghur, uzbek, vietnamese, welsh, xhosa, yiddish, yoruba and zulu. In many cases, the result for the backtranslation was the same for several languages, and sometimes identical to the original sentence. Therefore, a duplicate elimination process was required to avoid sample repetition in the training set.

In relation to Easy Data Augmentation, we detected some relevant issues with random insertions, deletions and swaps, that led to a performance decrease in the generated models. These were due to the nature of the problem at hand and were mainly related to the existence of some particular keywords (such as "help") that were randomly inserted or deleted changing the meaning of the sentence, but maintaining the label assigned to the original text despite that they clearly belonged to a different class after the operation. Therefore, we only applied a synonym replacement instead of the entire Easy Data Augmentation. In this case, we generated 4 new samples for every existing sentence in the original training set, according to the recommended usage parameters reported in [43]. Then, we replaced a fixed percentage of words by a synonym chosen at random, avoiding replacements of a list of common stop words.

In order to determine the optimum percentage, we evaluated several replacement rates and used the one leading to the maximum performance.

C. Construction of a Test Corpus

To be able to fairly evaluate intent classification and entity extraction results, a second corpus $\mathfrak{C}_2$ was built. The setting was identical to the one reported for the first experience described in Section IV-A, but involved 42 different students in the same age group, who had to solve a series of 3 algebraic word problems that were also different from the ones used in the first experience. Such a setting helps increase the independence between the training and the test set, and enable a more objective evaluation of the generated models.

From this experience, 5 711 messages from students were obtained, which turned into 1 973 after duplicate removal. Again, messages were translated into English by the research team and revised for correctness by a native English speaker and manually labelled, to construct an empirical test set that allowed us to evaluate the results in a realistic context.

V. Model Creation and Results

Using Rasa, we created an intent classification and entity recognition model by performing the following sequence of steps. First, we determined the most appropriate NLU pipeline for our particular case. Then, we sequentially configured the parameters of the synonym replacement and backtranslation mechanisms to optimize predictions and evaluate the improvement obtained by the proposed data augmentation methods.

A. Experimental Setting

For consistency reasons, we have adopted the same setting in all experiments reported in this section. For a sound evaluation, a validation set was extracted from the corpus $\mathfrak{C}_1$ described in section IV-A, containing 10% of the entries (629 utterances). The remaining 5 664 utterances were used to train the learning model, using the validation dataset to control the generalization error and decide when to stop training to avoid overfitting. Results reported are the ones obtained for the 1 973 sentences contained in the test corpus $\mathfrak{C}_2$ described in section IV-C, which were not seen at any stage of the model construction. This data composition is graphically illustrated in Fig. 4.

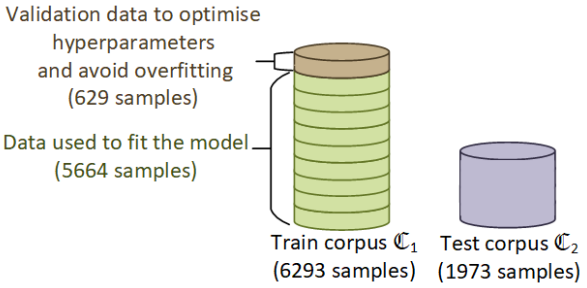

Fig. 4. Data composition used in the experiments.

B. Performance Assessment

To make decisions along the process, we took into consideration the particularities of our classification problem, which is highly imbalanced. This is because the dataset was obtained from real interactions, and therefore it contains significantly more samples from the most frequent intents and entities.

Typical measures to assess classification performance for a class C_i are precision, recall, and the so-called F_1-score. For a given class C_i, let's denote the number of samples that were correctly classified as members of this class as TP_i (true positives); the number of samples that were incorrectly assigned to the class as FP_i (false positives); the number of correctly classified samples into a class other than C_i as TN_i (true negatives); and the number of samples of class C_i that were assigned to a different class as FN_i (false negatives). Then, the precision (p_i) for the class C_i, indicates the proportion of positive identifications that were actually correct, and it is given by:

$$p_i = \frac{TP_i}{TP_i + FP_i}; \tag{1}$$

the recall measure indicates the proportion of positive instances that were correctly identified, and it is defined as

$$r_i = \frac{TP_i}{TP_i + FN_i}; \tag{2}$$

and the F_{1i}-score [44] is defined as the harmonic mean of the precision and recall, and can be computed as:

$$F_{1i} = \frac{2 \cdot (p_i \cdot r_i)}{p_i + r_i}. \tag{3}$$

The F_{1i}-score is generally considered as a good compromise between recall and precision and it is widely accepted as an evaluation measure in highly imbalanced problems like ours. Nevertheless, the F_{1i}-score is obtained for each class C_i and needs to be summarized into an easy-to-interpret single value by combining the results obtained for each class. This is done by using a weighted-average measure, which gives each class an importance that is proportional to its size and it is hence more appropriate for an imbalanced multi-class classification setting like ours. Assuming K different classes, the weighted-average of the F_1-scores is defined as

$$F_{1W} = \frac{\sum_{i=1}^{K} (TP_i + FN_i) \cdot F_{1i}}{\sum_{i=1}^{K} (TP_i + FN_i)}, \tag{4}$$

and has been adopted as a reference metric to evaluate the comparative performance of alternative models in this paper.

C. NLU Pipeline Selection

The objective of this stage is to find the pipeline that best fits our specific needs. This is a domain-dependent step that consists of building a set of alternative configurations and compare their performance in the specific application scenario.

In our particular case, 6 different pipelines were suggested for the comparison. The first configuration was the one provided by default. The rest were ad-hoc modifications proposed by careful observing the data collected and elaborating hypotheses addressed to improve the results in our particular use case.

The default configuration (see Fig. 5) is composed of standard pre-designed components and it is not bound to a specific language because it does not use any pre-trained word embeddings. When this configuration is used, tokens are produced by using a simple algorithm that splits on and discards whitespace characters (*WhitespaceTokenizer*). Then, the DIET classifier is fed with both token-based and whole-sentence based features. With regard to token-based features, the *LexicalSyntacticFeaturizer* builds a binary feature vector by using the previous, current and next token. Each vector

TABLE III
COMPARISON OF PROPOSED PIPELINE CONFIGURATIONS, IN TERMS OF THEIR MAJOR PROPERTIES

NLU Pipeline Configuration	Preprocessing of Messages	Pre-trained English Model	Character n-grams	Dense Featurizer	Intent Classifier	Entity Extractor
Default Configuration	✗	✗	$n = 1$ to 4	✗	DIET	DIET
Added pre-processing	✓	✗	$n = 1$ to 4	✗	DIET	DIET
Pre-trained for English	✓	✓	$n = 1$ to 4	✓	DIET	DIET
No unigrams	✓	✓	$n = 2$ to 4	✓	DIET	DIET
DIET+CRF	✓	✓	$n = 2$ to 4	✓	DIET	CRF
SVM+CRF	✓	✓	$n = 2$ to 4	✓	SVM	CRF

```
language: en
pipeline:
  - name: WhitespaceTokenizer
  - name: RegexFeaturizer
  - name: LexicalSyntacticFeaturizer
  - name: CountVectorsFeaturizer
  - name: CountVectorsFeaturizer
    analyzer: char_wb
    min_ngram: 1
    max_ngram: 4
  - name: DIETClassifier
    epochs: 100
    constrain_similarities: true
  - name: EntitySynonymMapper
  - name: ResponseSelector
    epochs: 100
    constrain_similarities: true
  - name: FallbackClassifier
    threshold: 0.3
    ambiguity_threshold: 0.1
```

Fig. 5. Configuration pipeline offered by Rasa.

value is assigned a 0 or a 1, depending on whether the tokens meet a given condition e.g., is a lowercase word, is the beginning of a sentence, is the end of a sentence. Whole sentence features in this configuration were produced by the *CountVectorFeaturizer*, which computes a bag-of-words representation of the sentence. Two different versions of this featurizer are used. The first one uses words to build the dictionary, and the second one considers character n-grams of size $1 \leq n \leq 4$. The *RegexFeaturizer* creates a list of regular expressions that are present in the training data and sets a feature for each regular expression, marking whether it was found in the user utterance.

Neither the *EntitySynomynMapper* or the *ResponseSelector* components have an effect. The former is used to help the entity mapper by specifying list of terms that refer to the same thing; and the latter is related to issuing a response utterance, which is outside the scope of the current system as it falls within the responsibilities of the ITS. Finally, the *FallbackClassifier* component instructs the conversational system not to output an intent when the confidence score is below 0.3, or when the difference between the two highest ranked intents is smaller than 0.1. In these cases, the user message is classified as a special intent known as *nlu_fallback*, causing the system to prompt the user to rephrase the message.

The first alternative configuration was generated by adding a custom component as the entry point of the pipeline, that pre-processed the original text by converting it to lowercase and forcing the appearance of a space character at each side of the "=" and ":" symbols. This operation helps the subsequent *WhitespaceTokenizer* to better separate tokens in utterances such as "x:the number of rabbits" or "total number of heads=y".

A second alternative configuration builds on the previous one and attempts to improve the results by focusing on the English language by using spaCy-based components [45]. In particular, the largest available model (en_core_web_lg) was used; the *WhitespaceTokenizer* was replaced by the *SpacyTokenizer*, which is also based on white spaces but also takes into account punctuation and language-dependent special case rules e.g. to divide "don't" into two tokens; and the *SpacyFeaturizer* was added to generate dense features for each token and also for the entire sentence (a meaning-related one-dimensional vector of a fixed-size).

The third alternative pipeline considers character n-grams of size $2 \leq n \leq 4$ to build the bag-of-words representation of the sentence. The intention behind this modification was to limit the effect of single-letter quantity names on the classification of the intent.

The last two configurations aimed at evaluating the use of alternative classifiers for both intent classification and entity extraction. Despite the general higher performance of the DIET classifier at both tasks, and hence the Rasa recommendation to try it as the first option, we tried other alternatives to ensure that the higher performance of DIET also holds in our specific setting. In the first attempt, we kept DIET as the intent classification option but used *CRFEntityExtractor* for entity recognition. As the last option, we also replaced the DIET classifier by *SklearnIntentClassifier*. *CRFEntityExtractor* uses Conditional Random Fields (CRF) [46], a popular statistical model that can be thought of as an undirected Markov chain where the time steps are words and the states are entity classes; and *SklearnIntentClassifier* trains a linear Support Vector Machine (SVM) [47] , using a grid search to fine-tune the hyperparameters C and γ.

Table III summarizes the major characteristics of each proposed pipeline. To compare their performance in our particular context, we used the experimental setting described in Section V-A. Fig. 6 shows the results for both intent classification and entity recognition in terms of the weighted F_1-score, computed according to Eq. 4. As can be observed, the best results are obtained for the solution that incorporates the proposed

TABLE IV
NUMBER OF SAMPLE UTTERANCES FOR EACH SYNONYM REPLACEMENT PERCENTAGE

Synonym replacement percentage	0%	5%	10%	15%	20%	25%	30%	35%	40%	60%	80%
Number of utterances	6 293	16 073	16 081	16 152	16 226	16 250	16 273	16 298	16 371	16 405	16 424

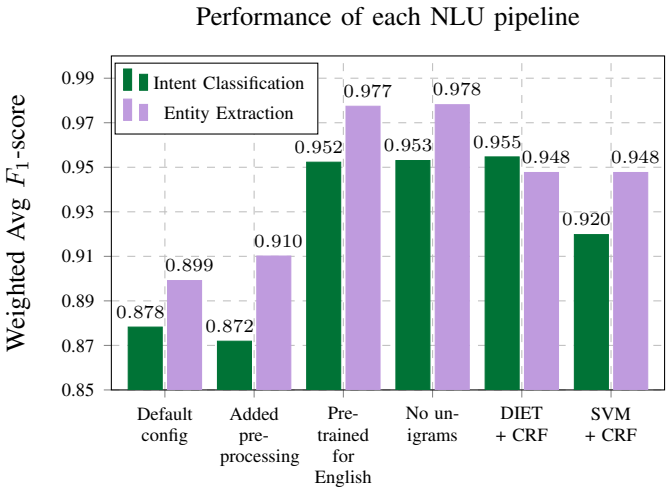

Fig. 6. Classification performance of each NLU pipeline.

pre-processing step, uses the pre-trained English model and discards single-character unigrams. In general, each of these measures progressively increase the intent classification and entity extraction performance, reaching weighted F_1-scores of 0.953 and 0.978, respectively when they are jointly applied. On the contrary, the use of a linear SVM for intent classification or/and CRF for entity recognition both have a negative impact on performance. The small decrease observed in intent prediction for the added pre-processing option was mainly due to the use of the "=" and ":" characters as token separators. This helped entity extraction in letter definitions such as "x:age of Ann" but at the same time provoked that equations such as "x=x+1" were split into several tokens, causing confusion between equations, expressions and quantity definitions, and activating the *FallbackClassifier* in some infrequent occasions. This uncertainty does not happen when features are used in combination with an English language embedding that helps disambiguate between these intents by recognizing meaningful words.

D. Configuration of Data Augmentation

Once the NLP pipeline that best fitted our purpose was selected, the data augmentation methods were also configured to optimize the network performance.

The original train corpus $\mathfrak{C}_1$ was first augmented by using a varying percentage of synonyms. 4 new sentences were generated for every original sentence, and the percentage of words replaced by a synonym was considered a parameter that needed to be fine-tuned. Different percentages were attempted, namely: 0%, 5%, 10%, 15%, 20%, 25%, 30%, 35%, 40%, 60% and 80%. The resulting number of utterances for each replacement rate are shown in Table IV. Note that differences are due to the elimination of duplicates, which has a lesser impact as the number of replacements grows.

Fig. 7 shows the results in terms of the weighted F_1-score, when fitting the model with the augmented corpus and evaluating using the test corpus $\mathfrak{C}_2$. Although performance differences are relatively small, the best results were obtained for a 10% replacement rate and hence we fixed this percentage for the synonym replacement step. While the optimal replacement rate is indeed application-dependent, the generation of 4 new samples for every utterance in the original training set and 10% synonym replacement seems a good general recommendation, in view of our results and the ones reported in [43] on a variety of datasets.

TABLE V
NUMBER OF TRAINING UTTERANCES FOR EACH COMBINATION OF DATA AUGMENTATION SCHEMES

Data augmentation scheme	Samples
No augmentation	6 293
Backtranslation	167 246
10% Synonym replacement	16 081
Backtranslation + 10% Synonym replacement	365 122

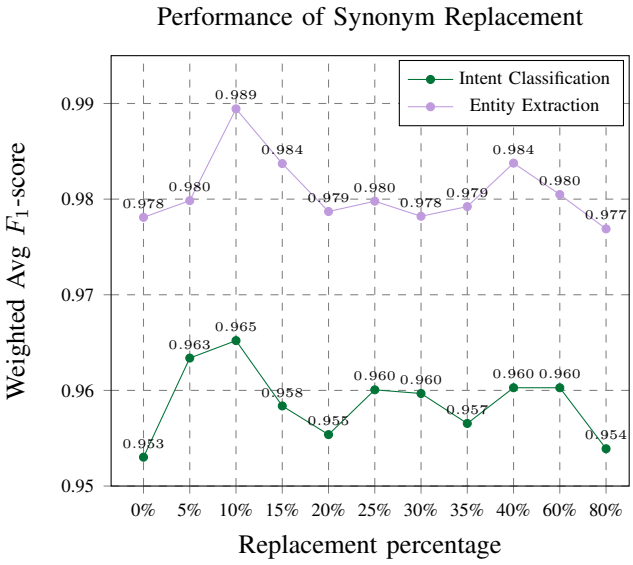

Fig. 7. Classification performance as a function of the percentage of synonym replacement.

Next, we examined the potential benefit of the backtranslation method, when applied both in isolation and jointly with a 10% synonym replacement. Fig. 8 shows the results for all possible four combinations of both methods. While synonym replacement always has a beneficial effect on both intent classification and entity recognition, the joint application with backtranslation has a negative effect on both variables. This is despite the significant increase in the number of training samples (see Table V). We attribute this effect to the specific characteristics of our particular domain, which is linked to mathematical concepts whose sense is frequently

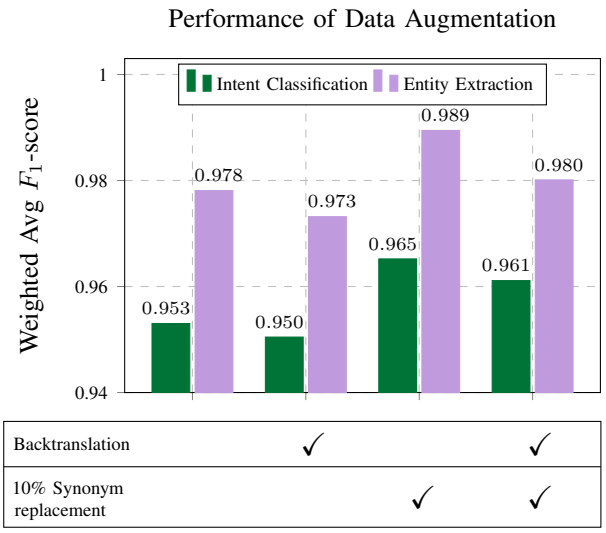

Fig. 8. Weighted F_1-scores for different data augmentation approaches.

lost in the backtranslation process. For example, the utterance "new variable" yields some entries such as "new version", "new edit" or "new update", which are provided as training data under the inappropriate intent label "define letter question". As this negative contribution may be domain-dependent, we strongly recommend not extrapolating these results and carefully testing the effect of the backtranslation approach when creating other conversational agents, to determine the best combination of data augmentation schemes.

VI. Performance as a Function of the Train Set Size

Another interesting analysis consists of studying how the performance of the selected configuration varies when increasing the amount of training data. Although it is expected to increase with the number of available samples, this study attempts to find out the relation between size and performance, to help determine the impact of acquiring more data and provide a general indication about the minimum amount of information required to achieve an acceptable behaviour. To this end, we created 10 different models using the corpus $\mathfrak{C}_1$, progressively increasing the train set with samples that were not included in the previous model. For each new model, 10% of the corpus data were added, until using the entire repository. The results reported are the ones obtained for the 1 973 sentences contained in the corpus $\mathfrak{C}_2$, reserving 10% of the train data for validation in all cases.

Fig. 9 shows the results for both intent classification and entity extraction for the best combination of data augmentation strategies reported in Section V-D, namely generation of 4 new samples per original utterance with a 10% synonym replacement. These are again expressed in terms of the weighted F_1-score. Although they both show a logarithmic dependency with the train set size, we can observe that intent classification has a more steep curve, especially for small values.

The relatively high weighted F_1-scores obtained even for small quantities of train data support the use of the Rasa conversational framework to help adapt existing tutoring systems to a natural language-based interaction. The first value in the graph corresponds to an augmented training set generated by using just 629 sample utterances from $\mathfrak{C}_1$, and the system is already able to distinguish between 25 intents with a weighted F_1-score above 0.88, rapidly increasing to a value above 0.95 when the system is trained with 30% of the train data (1 888 original utterances). Results are even better for entity recognition, with a weighted F_1-score above 0.95 even for the smallest training size.

The graph also shows that intent classification gets fairly stable when the augmented training set is generated by using more than 2 517 original utterances (40% of the total), while entity recognition seems to keep slowly improving for larger sizes. In general, both graphs seem to describe a logarithmic growth, increasing rapidly for small train sizes and soon levelling off to become asymptotic. The relatively high performance reached for the complete train set supports the validity of the approach in practical settings, opening new possibilities for the easy development of conversational platforms that provide a more convenient form of interaction, especially in educational environments.

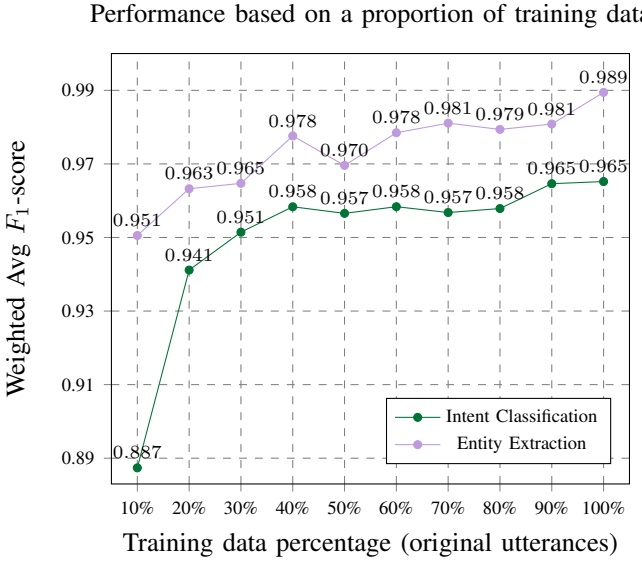

Fig. 9. Intent classifier and Entity extractor performance based on a proportion of training data.

d,,id,d,,id,

VII. Generalization of the Approach

By using a specific case study, we have shown that it is possible to adapt an ITS to a conversational interface with a reasonable effort, using already available open-source technology. Other available frameworks provide different performances but share the same working principles based on the identification of intents and entities. Some commercial products such as Amazon Lex [8] are already powered by improved language models that increase their productivity, and advances in this field are rapidly happening. In this direction, the fast development of advanced language models such as GPT-3 [48] has enabled multiple new applications of natural language processing technology, e.g. automatic coding.

The adoption of such models by conversational frameworks will indeed contribute to improving their overall performance, having a positive effect on increasing the use of natural language as a means of interaction. Despite these expected future gains, the accuracy results reported in this paper support the current use of available software for the construction of C-ITS.

Design guidelines provided in this paper can easily be exported to other problems and subject areas, and are aligned with recent development in the natural language processing field. For example, related recent approaches have been used in [49] to train a sentiment analysis tool, and in [50] to create a chatbot focused on answering questions, handling complaints and supporting transactions, both using the Rasa conversational framework. These works share the same methodology to determine the most effective pipeline, which indeed strongly depends on the application area, as it is implied by the substantially different results reported ind,,id,[49], [50]d,,id,[49], [50] [49], [50]. Still, the guidelines need to be adapted to each specific case study, according to intermediate results. For example, if no further gains are observed by using data augmentation methods, dataset corrections should usually be avoided, to produce a more generalizable model that is able to cope with existing mistakes in the text messages. Another important factor to take into account is the size of the training corpus. Due to the scarce number of possible actions in our case study, the size of this dataset and the diversity introduced by the more than 80 individuals who participated in the data collection were considered sufficient to make a first attempt to produce a reliable system. However, we can even sense from the plots in Fig. 9 that there is room for improvement by increasing the number of samples in the training set, especially in the case of entity extraction, which still shows an upward trend. Retraining the system at a future stage using the data already collected when the ITS is in production is always an option to consider. Like most other conversational toolkits, Rasa also provides the Rasa X web interface tool, which lets the designer revise the intent classification results and manually fix incorrect entries or click on a checkmark to reinforce that behaviour.

The migration to a conversation interface was relatively easy in our case study because of the particular architecture of the original button-based version, which used a typical client-server model and placed most of the controller logic on the client. In our particular setting, the knowledge base was fully located at the back-end, including the domain, instruction and student models. Following a typical architecture commonly adopted in other works, e.g. [51], [52], most user actions were processed by sending requests to the back-end, attaching data that the user had filled in the graphical user interface. The integration of the conversational agent just required the implementation of a new simpler interface, along with minor adjustments on the server-side component and a slight modification of the general application architecture, which is shown in Fig. 10. When the user enters a new utterance, it is sent to the implemented Rasa service, which returns the corresponding intent and entities extracted from the message. The output from the Rasa services is then used to build an identical HTTP request as it was used in the existing button-based version. Finally, the request is sent to the Back-end, to update the models in the same way as it was done in the button-based user interface.

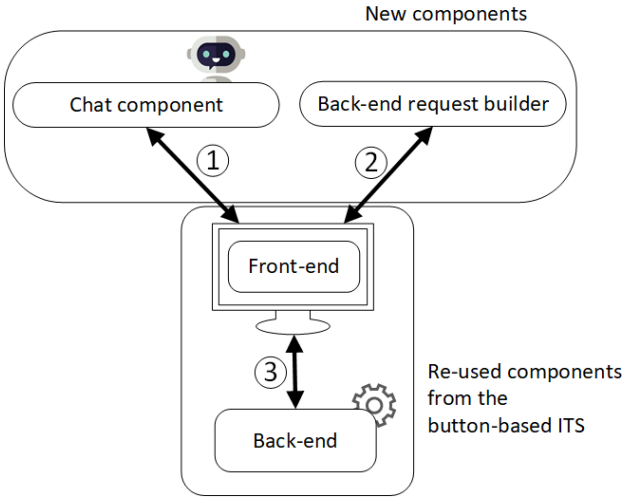

Fig. 10. Required modifications in the architecture of the existing ITS.

To decide on the potential use of a similar scheme in other existing systems, we should carefully analyse the structure of the original application. While changes were relatively easy in our case study because they were mainly applied on the client side, they may have substantially higher complexity in other thin client architectures that place most of the programming logic on the server side and send an updated web page as a response to client requests. In general, the adaptation difficulty will increase when there is a strong dependency between the graphical user interface and the application logic, and decrease in the case of loosely coupled architectures in which the user interface appears as an independent component.

VIII. Conclusions and Future Work

The advantages of using natural language interfaces in Intelligent Tutoring Systems extend beyond being a more convenient form of interaction. In addition to reported educational gains [2], they also provide a valuable opportunity to offer unsensorized affective support based on the analysis of the user utterances, with the potential to surpass the benefits of existing proposals, e.g. [53].

In this paper, we have evaluated the potential of conversational frameworks to underpin the development of new-generation ITS that interact with the user by using natural language. A pilot study based on an existing ITS called HBPS [11], [12], [13] has shown the suitability of such platforms for the task, at the same time as it has served to establish some methodological guidelines for new developments or adapting existing ITS to support a natural language interaction. There are major differences between our work and others related to the development of the conversational agents mentioned in Section II [32], [33], [34], [35], [36], [37]. We have used an open-source tool and focused on the generalization of the suggested method. The proposed methodological

guidelines include a separate evaluation of the conversational agent, facing both the pipeline design and the use of data augmentation methods to improve the system's performance. In this paper, the suggested approach has been applied to one specific case study which is particularly challenging, because of the presence of expressions and other mathematical elements in user utterances.

The successful results obtained as part of this research are clearly in support of this kind of technology, although we must remark that there are still open issues that should be further investigated. In the first place, and despite that the weighted F_1-scores are a fair and objective measure of performance, there are other practical considerations that affect the global performance of the system. For example, the impact of a misclassification must be evaluated from not only an educational perspective but also from a usability point of view. Whether a misclassification results in a slightly inappropriate or unexpected answer or produces inconsistent behaviour with no easy recovery is an essential aspect. This issue may be quickly evaluated by using the confusion matrix obtained from the test data, to analyse whether inappropriate answers may be issued as a response to a user's utterance. We plan to use such an analysis not only to detect potential sources of usability problems but also to avoid them by providing alternative mechanisms that are able to warn of potentially wrong predictions. Another possible improvement consists of merging or separating intents to help classification accuracy. On the one hand, decreasing the number of intents generally has a beneficial effect on performance caused by the reduction of the number of classes. On the other hand, increasing it also has a positive effect on the system's flexibility, allowing the designer to provide more adapted responses according to the predicted intent. Further research is needed to achieve the right balance between performance and response adaptability, by providing a set of common guidelines and procedures that can be used to drive the intent identification task.

Another relevant aspect concerns the relative importance of intent classification and entity recognition. In an education system like ours, whose purpose is to understand the user's message to let the system issue an appropriate response, it is the intent that guides the conversation. Nevertheless, we cannot disregard the existing relationship between intents and entities. They are complementary, and immediate future work will focus on considering inter-dependencies to improve classification results. These inter-dependencies are application-specific, but especially relevant when an intention is necessarily associated with entities that must or cannot co-exist with it. At the moment, Rasa provides an option to select what entities to report for each intent and discard the rest. Nevertheless, this functionality does not exploit existing relations between entities and intents, which could be seamlessly used to detect potentially invalid intent classifications or reinforce certain decisions, according to whether the entities detected match the expected ones. Such an approach would help re-programming a more effective *FallbackClassifier* component that assists intent classification when the confidence score is below the established threshold or when the difference between the two highest-ranked intents is smaller than the ambiguity threshold.

We shall neither disregard language-related aspects. In our experience, the English versions of language models usually work better and yield more accurate results. However, a systematic evaluation of the influence of language on the performance of conversational systems is still pending. While the English language is supported by models and embeddings provided as part of standard libraries such as spaCy [45], other less used languages do not have the same kind of support. Although this argument holds true at this moment, NLP tools are rapidly evolving and recent developments in related technologies, e.g. translation, will ease the generation of exhaustive models for even minority languages.

It is also worth commenting on common functionalities that conversational frameworks provide for continuous improvement. Most of them offer tools that allow the designer to supervise and improve the conversational system in a simple and effective way, by keeping track of all previous classifications made. These classification results can easily be revised at a glance, and mistakes can easily be tagged so that the system is re-trained by making special emphasis on the marked items. New efforts are required to further automate this task, e.g., by discovering inconsistent interactions with the aim to offer the designer already filtered incorrect decisions.

Finally, we should remark on the importance of considering the implications of the subject area in the adaptation process. While natural language interfaces provide undoubtful benefits in some cases, we shall analyse the consequences of adopting such a way of interaction in each particular domain. Other than the required coding effort, migrating to a C-ITS may also need additional changes related to how materials are presented, when aids are given and other instructional issues. These implications should be carefully studied before starting the adaptation, ideally counting on experts in different disciplines to evaluate the results from both a usability and a learning perspective.

IX. Acknowledgements

This research has been supported by project PGC2018-096463-B-I00, funded by MCIN/AEI/10.13039/501100011033 and "ERDF A way of making Europe"; and project TED2021-129485B-C41 by the Ministry of Science and Innovation (Recovery, Transformation and Resilience Plan).

References

[1] M. Tomasello, A. C. Kruger, and H. H. Ratner, "Cultural learning," *Behavioral and Brain Sciences*, vol. 16, no. 3, p. 495–511, 1993.

[2] J. Paladines and J. Ramírez, "A systematic literature review of intelligent tutoring systems with dialogue in natural language," *IEEE Access*, vol. 8, pp. 164 246–164 267, 2020.

[3] B. Nye, A. Graesser, and X. Hu, "Autotutor and family: A review of 17 years of natural language tutoring," *International Journal of Artificial Intelligence in Education*, vol. 24, no. 4, pp. 427–469, 2014.

[4] K. VanLehn, R. Freedman, P. Jordan, C. Murray, R. Osan, M. Ringenberg, C. Rosé, K. Schulze, R. Shelby, D. Treacy, A. Weinstein, and M. Wintersgill, "Fading and deepening: The next steps for andes and other model-tracing tutors," in *Intelligent Tutoring Systems*, G. Gauthier, C. Frasson, and K. VanLehn, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2000, pp. 474–483.

[5] C. D'Helon, J.-W. Ahn, V. Kasireddy, and N. Mukhi, "Interactive learning in a conversational intelligent tutoring system using student feedback, concept grouping and text linking," in *INTED2019 Proceedings*, ser. 13th International Technology, Education and Development Conference. IATED, 2019, pp. 2820–2830.

[6] I. Lauriola, A. Lavelli, and F. Aiolli, "An introduction to deep learning in natural language processing: Models, techniques, and tools," *Neurocomputing*, vol. 470, pp. 443–456, 2022.

[7] Dialogflow. https://dialogflow.com/ (accessed 27 January 2022).

[8] Amazon lex – build conversation bots. https://aws.amazon.com/lex/ (accessed 27 January 2022).

[9] Ibm watson. https://www.ibm.com/watson (accessed 27 January 2022).

[10] T. Bocklisch, J. Faulkner, N. Pawlowski, and A. Nichol, "Rasa: Open source language understanding and dialogue management," 2017.

[11] M. Arevalillo-Herraez, D. Arnau, and L. Marco-Giménez, "Domain-specific knowledge representation and inference engine for an intelligent tutoring system," *Knowledge-Based Systems*, vol. 49, pp. 97–105, 2013.

[12] D. Arnau, M. Arevalillo-Herraez, L. Puig, and J. A. González-Calero, "Fundamentals of the design and the operation of an intelligent tutoring system for the learning of the arithmetical and algebraic way of solving word problems," *Computers & Education*, vol. 63, pp. 119–130, 2013.

[13] D. Arnau, M. Arevalillo-Herráez, and J. A. González-Calero, "Emulating human supervision in an intelligent tutoring system for arithmetical problem solving," *IEEE Transactions on Learning Technologies*, vol. 7, no. 2, pp. 155–164, 2014.

[14] C. Benzmüller, H. Horacek, I. Kruijff-Korbayová, M. Pinkal, J. Siekmann, and M. Wolska, "Natural language dialog with a tutor system for mathematical proofs," in *Cognitive Systems*, R. Lu, J. H. Siekmann, and C. Ullrich, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007, pp. 1–14.

[15] V. Aleven, A. Ogan, O. Popescu, C. Torrey, and K. Koedinger, "Evaluating the effectiveness of a tutorial dialogue system for self-explanation," vol. 3220, 2004, pp. 443–454.

[16] N. Heffernan, K. Koedinger, and L. Razzaq, "Expanding the model-tracing architecture: A 3rd generation intelligent tutor for algebra symbolization." *I. J. Artificial Intelligence in Education*, vol. 18, pp. 153–178, 2008.

[17] C. Rosé, P. Jordan, M. Ringenberg, S. Siler, K. Vanlehn, and A. Weinstein, "Interactive conceptual tutoring in atlas-andes," 2001.

[18] V. Rus, S. D'Mello, X. Hu, and A. Graesser, "Recent advances in conversational intelligent tutoring systems," *Ai Magazine*, vol. 34, pp. 42–54, 2013.

[19] L. Benotti, M. C. Martnez, and F. Schapachnik, "A tool for introducing computer science with automatic formative assessment," *IEEE Transactions on Learning Technologies*, vol. 11, no. 2, pp. 179–192, 2018.

[20] O. Alobaidi, K. Crockett, J. O'Shea, and T. Jarad, "Abdullah: An intelligent arabic conversational tutoring system for modern islamic education," *Lecture Notes in Engineering and Computer Science*, vol. 2, pp. 762–768, 2013.

[21] Z. Cai, A. Graesser, C. Forsyth, C. Burkett, K. Millis, P. Wallace, D. Halpern, and H. Butler, "Trialog in aries: User input assessment in an intelligent tutoring system," 2011.

[22] C. Wolfe, V. Reyna, C. Widmer, E. Cedillos-Whynott, P. Brust-Renck, A. Weil, and X. Hu, "Understanding genetic breast cancer risk: Processing loci of the brca gist intelligent tutoring system," *Learning and Individual Differences*, vol. 49, pp. 178–189, 2016.

[23] M. McTear, *Conversational AI: Dialogue Systems, Conversational Agents, and Chatbots*, ser. Synthesis Lectures on Human Language Technologies. Morgan & Claypool, 2020, vol. 13.

[24] Y. Zhang, R. Jin, and Z.-H. Zhou, "Understanding bag-of-words model: A statistical framework," *International Journal of Machine Learning and Cybernetics*, vol. 1, pp. 43–52, 2010.

[25] A. C. Graesser, K. Wiemer-Hastings, P. Wiemer-Hastings, and R. Kreuz, "Autotutor: A simulation of a human tutor," *Cognitive Systems Research*, vol. 1, no. 1, pp. 35–51, 1999.

[26] E. M. Cedillos-Whynott, C. R. Wolfe, C. Widmer, P. G. Brust-Renck, A. M. Weil, and V. F. Reyna, "The effectiveness of argumentation in tutorial dialogues with an intelligent tutoring system for genetic risk of breast cancer," *Behavior Research Methods*, vol. 48, pp. 857–868, 2016.

[27] A. Latham, K. Crockett, and D. McLean, "An adaptation algorithm for an intelligent natural language tutoring system," *Computers & Education*, vol. 71, pp. 97–110, 2014.

[28] N.-T. Le and L. Wartschinski, "A cognitive assistant for improving human reasoning skills," *International Journal of Human-Computer Studies*, vol. 117, pp. 45–54, 2018.

[29] S. Aljameel, J. O'Shea, K. Crockett, A. Latham, and M. Kaleem, "Lana-i: An arabic conversational intelligent tutoring system for children with asd," in *Intelligent Computing*, R. Arai, Koheiand Bhatia and S. Kapoor, Eds. Cham: Springer International Publishing, 2019, pp. 498–516.

[30] A. Olney, N. Person, and A. Graesser, "Guru: Designing a conversational expert intelligent tutoring system," in *Cross-Disciplinary Advances in Applied Natural Language Processing: Issues and Approaches*, B.-D. Chutima, P. M. McCarthy, and T. Lamkin, Eds. Hershey, PA, USA: IGI Global, 2012, pp. 156–171.

[31] B. D. Nye, P. I. J. Pavlik, A. Windsor, A. M. Olney, M. Hajeer, and X. Hu, "Skope-it (shareable knowledge objects as portable intelligent tutors): overlaying natural language tutoring on an adaptive learning system for mathematics," *International Journal of STEM Education*, vol. 5, no. 1, 2018.

[32] H. Agus Santoso, N. Anisa Sri Winarsih, E. Mulyanto, G. Wilujeng saraswati, S. Enggar Sukmana, S. Rustad, M. Syaifur Rohman, A. Nugraha, and F. Firdausillah, "Dinus intelligent assistance (dina) chatbot for university admission services," in *2018 International Seminar on Application for Technology of Information and Communication*, 2018, pp. 417–423.

[33] H. T. Hien, P.-N. Cuong, L. N. H. Nam, H. L. T. K. Nhung, and L. D. Thang, "Intelligent assistants in higher-education environments: The fit-ebot, a chatbot for administrative and learning support." New York, NY, USA: Association for Computing Machinery, 2018.

[34] K. Katchapakirin and C. Anutariya, "An architectural design of scratchthai: A conversational agent for computational thinking development using scratch." New York, NY, USA: Association for Computing Machinery, 2018.

[35] A. D. Topal, C. D. Eren, and A. K. Geçer, "Chatbot application in a 5th grade science course," *Education and Information Technologies*, vol. 26, pp. 6241–6265, 2021.

[36] N. González-Castro, P. J. Muñoz-Merino, C. Alario-Hoyos, and C. Delgado Kloos, "Adaptive learning module for a conversational agent to support mooc learners," *Australasian Journal of Educational Technology*, vol. 37, no. 2, p. 24–44, 2021.

[37] J. Paladines, J. Ramírez, and M. Berrocal-Lobo, "Integrating a dialog system with an intelligent tutoring system for a 3D virtual laboratory," *Interactive Learning Environments*, vol. 0, no. 0, pp. 1–14, 2021.

[38] T. Bunk, D. Varshneya, V. Vlasov, and A. Nichol, "DIET: lightweight language understanding for dialogue systems," 2020.

[39] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Minneapolis, Minnesota: Association for Computational Linguistics, 2019, pp. 4171–4186.

[40] J. Pennington, R. Socher, and C. Manning, "GloVe: Global vectors for word representation," in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Doha, Qatar: Association for Computational Linguistics, 2014, pp. 1532–1543.

[41] M. Henderson, I. Casanueva, N. Mrkšić, P.-H. Su, T.-H. Wen, and I. Vulić, "ConveRT: Efficient and accurate conversational representations from transformers," in *Findings of the Association for Computational Linguistics: EMNLP 2020*. Online: Association for Computational Linguistics, 2020, pp. 2161–2174.

[42] R. Sennrich, B. Haddow, and A. Birch, "Improving neural machine translation models with monolingual data," in *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*. Berlin, Germany: Association for Computational Linguistics, 2016, pp. 86–96.

[43] J. Wei and K. Zou, "EDA: Easy data augmentation techniques for boosting performance on text classification tasks," in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Hong Kong, China: Association for Computational Linguistics, 2019, pp. 6382–6388.

[44] M. Sokolova and G. Lapalme, "A systematic analysis of performance measures for classification tasks," *Information Processing & Management*, vol. 45, no. 4, pp. 427–437, 2009.

[45] M. Honnibal and I. Montani, "spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing," vol. 7, no. 1, 2017, to appear.

[46] J. D. Lafferty, A. McCallum, and F. C. N. Pereira, "Conditional random fields: Probabilistic models for segmenting and labeling sequence data," in *Proceedings of the Eighteenth International Conference on Machine Learning*, ser. ICML '01. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2001, pp. 282–289.

[47] C.-C. Chang and C.-J. Lin, "Libsvm: A library for support vector machines," *ACM Trans. Intell. Syst. Technol.*, vol. 2, no. 3, 2011.
[48] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. M. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei, "Language models are few-shot learners," 2020. [Online]. Available: https://arxiv.org/abs/2005.14165
[49] M. Arevalillo-Herráez, P. Arnau-González, and N. Ramzan, "On adapting the DIET architecture and the Rasa conversational toolkit for the sentiment analysis task," *IEEE Access*, vol. 10, pp. 107 477–107 487, 2022.
[50] L. W. Narendra and E. R. Setyaningsih, "Designing a transactional smart assistant in indonesian using rasa framework," in *2021 7th International Conference on Electrical, Electronics and Information Engineering (ICEEIE)*, 2021, pp. 1–6.
[51] S. Aljameel, J. O'Shea, K. Crockett, A. Latham, and M. Kaleem, "Lana-i: an arabic conversational intelligent tutoring system for children with asd," in *Intelligent Computing-Proceedings of the Computing Conference*. Springer, 2019, pp. 498–516.
[52] J. Paladines, J. Ramírez, and M. Berrocal-Lobo, "Integrating a dialog system with an intelligent tutoring system for a 3d virtual laboratory," *Interactive Learning Environments*, pp. 1–14, 2021. [Online]. Available: https://doi.org/10.1080/10494820.2021.1972012
[53] M. Arevalillo-Herráez, L. Marco-Giménez, D. Arnau, and J. A. González-Calero, "Adding sensor-free intention-based affective support to an intelligent tutoring system," *Knowledge Based Systems*, vol. 132, pp. 85–93, 2017.

Romina Soledad Albornoz-De Luise received the BS degree in mathematics from the National University of La Plata, in 2020. She is a doctoral student in Information technology, Communications and Computing, in the Escola Tècnica Superior d'Enginyeria at the University of Valencia. She has been an assistant professor for 6 years in different subjects: Data Analysis, Mathematical Analysis and Algebra, with the National University of La Plata. Her current research focuses on Intelligent Tutoring Systems and Natural Language Processing.

Miguel Arevalillo-Herráez received the first degree in Computing from the Technical University of Valencia, Spain, in 1993; the BSc in Computing from Liverpool John Moores University, UK, in 1994; and the PgCert in Teaching and Learning in Higher Education and the PhD degree in 1997, both also from Liverpool John Moores University, UK. In this same year, he gained accreditation as a teacher in HE by the Staff and Educational Development Association (SEDA) and became a senior lecturer at Liverpool John Moores University. In 1999, he left to work for private industry for a one-year period and came back to academy in 2000. He was the program leader for the Computing and Business degrees at the Mediterranean University of Science and Technology until 2006. Since then, he works as a lecturer at the University of Valencia (Spain), currently as a Full Professor. His research now concentrates on Intelligent Tutoring Systems, Educational Technology and Applied Artificial Intelligence.

David Arnau received the degree in Physics from the University of Valencia (Spain). He taught mathematics and physics at the secondary level for thirteen years. In 2010, he received a PhD in Educational Mathematics from the University of Valencia. At present, he works as a Full Professor at the Department of Didactics of Mathematics of the University of Valencia. He is a member of the board of the Spanish Society of Research in Mathematics Education (SEIEM). His research interest is centred in educational algebra and the development of interactive learning environments for the teaching and learning of word problem-solving.