information

Article

Multi-Microworld Conversational Agent with RDF Knowledge Graph Integration

Gabriel Boroghina [1], Dragos Georgian Corlatescu [1] and Mihai Dascalu [1,2,*]

[1] Computer Science & Engineering Department, University Politehnica of Bucharest, 313 Splaiul Independentei, 060042 Bucharest, Romania
[2] Academy of Romanian Scientists, Str. Ilfov, Nr. 3, 050044 Bucharest, Romania
* Correspondence: mihai.dascalu@upb.ro

Abstract: We live in an era where time is a scarce resource and people enjoy the benefits of technological innovations to ensure prompt and smooth access to information required for our daily activities. In this context, conversational agents start to play a remarkable role by mediating the interaction between humans and computers in specific contexts. However, they turn out to be laborious for cross-domain use cases or when they are expected to automatically adapt throughout user dialogues. This paper introduces a method to plug in multiple domains of knowledge for a conversational agent localized in Romanian in order to facilitate the extension of the agent's area of expertise. Furthermore, the agent is intended to become more domain-aware and learn new information dynamically from user conversations by means of a knowledge graph acting as a network of facts and information. We ensure high capabilities for natural language understanding by proposing a novel architecture that takes into account RoBERT-contextualized embeddings alongside syntactic features. Our approach leads to improved intent classification performance (F1 score = 82.6) when compared with a basic pipeline relying only on features extracted from the agent's training data. Moreover, the proposed RDF knowledge representation is confirmed to provide flexibility in storing and retrieving natural language entities, values, and factoid relations between them in the context of each microworld.

Keywords: conversational agents; intent classification; language models; syntactic parsing; microworlds; knowledge representation

Citation: Boroghina, G.; Corlatescu, D.G.; Dascalu, M. Multi-Microworld Conversational Agent with RDF Knowledge Graph Integration. *Information* **2022**, *13*, 539. https://doi.org/10.3390/info13110539

Academic Editors: Tudor Groza and Ryutaro Ichise

Received: 10 September 2022
Accepted: 11 November 2022
Published: 15 November 2022

1. Introduction

Conversational agents have quickly become one of the most helpful technologies at the crossroads of natural language processing and human–computer interaction, targeting the response to people's needs in terms of prompt, interactive, and easy-to-use ways for accessing an information technology service or controlling smart devices [1]. While intelligent devices and services, together with machine learning models, keep evolving, the future will bring computers closer and closer to our everyday lives. As such, we continue toward the goal of interacting with computers through natural language.

A universally knowledgeable agent would imply an impractically large variety of training examples and an AI agent capable of dealing with more complex inferences and context switching. Hence, conversational agents are generally developed and trained for individual domains and use cases, as exemplified in our previous practical study [2]. This implies that a specific set of user intents can be defined to cover the domain of knowledge for which the agent is designed. In this paper, we propose an extension to this rationale by allowing multiple unrelated fields to be fitted into the same agent while trying to reduce the impact on the overall agent performance. The current work is designed to support the feasibility of a multi-task agent suitable for services such as personal assistants which does not specifically target chatbot AIs leveraged by humanoid robots.

A significant amount of research interest has been shown in the domain of chatbots and conversational agents in recent years [3], with a consistent number of published papers

(a few thousand) that introduced methods, practical implementations, and evaluations, as well as surveys on existing work and the utility of such software systems. Recent surveys of the state of the art in the field of open-domain conversational AI [3,4] revealed that there is active development in bringing newer and more powerful NLP models and chatbots as a whole to reality.

An agent must store and have access to knowledge to be useful in practical environments requiring contextualized interactions with the user. In contrast to relational databases that have a rigid structure, the agent should support a flexible and easily extensible representation model capable of making inferences about the underlying information. As such, we find it optimal to represent agent data in a knowledge graph that provides a unified form for storing information as nodes and properties, which can be queried later by starting from any given entity and can be used to make new inferences. Therefore, we consider a native Resource Description Framework (RDF) triplestore [5] as a knowledge base for storing the various static and dynamic information the agent may harness and expand through user interactions. We describe a data representation model for various kinds of information and evaluate its viability by putting it into a practical personal assistant implementation with three microworlds.

The idea of using chatbot systems in conjunction with knowledge has proven valuable both in real-life applications and in research. For example, Ait-Mlouk and Jiang [6] explored the possibilities of integrating new knowledge bases in the form of a knowledge graph to a chatbot that can then be trained to provide information on the related subject. In general, chatbots target a specific field, and there is substantial work in the domain of public services for providing personalized information to citizens [7,8], medicine [9,10], movie recommendations [11], and museum helpers [12,13].

Our research objective is to introduce a personal assistant built for the Romanian language that concurrently supports multiple microworlds and interaction scenarios. Concurrently, we improve the performance of intent classification using a novel processing pipeline, custom language models, and syntactic features. In addition, we integrate RDF knowledge graphs to increase domain awareness and dynamically store new user information.

The main contributions of this paper are as follows:

- Introduce a personal assistant built for the Romanian language on top of the RASA framework that supports concurrently multiple microworlds, each targeting a different domain and in which additional microworlds can easily be plugged in;
- Propose a novel processing pipeline, custom language models, and syntactic features to improve intent classification. This approach is, to the best of our knowledge, a unique take on the chatbot's architecture and can be further used to improve its performance;
- Consider integrating RDF knowledge graphs with the conversational agent to ensure the extensibility of data modeling.

We made our code and dataset open source on GitHub [14] and Huggingface [15], respectively.

2. Literature Review

Recent machine learning models for generating natural language responses to users' utterances can be classified as retrieval-based (extracting relevant information from existent corpora), generative (producing tokens through decoders, given a dialogue context), or hybrid (e.g., retrieve and refine). The first category has the advantage of the responses being of higher quality (not synthetic) and consistent, and these models can be applied with more confidence in industry-oriented use cases. The generative approach can deliver more diverse and original responses based on the configured hyperparameters of the generative model, making these models more suitable for open-domain conversations (e.g., social chatbots). There is also the approach of gathering the responses of multiple models for a given user input and selecting one of them as a result. Another method, proposed by Hardalov et al. [16], leverages a machine reading comprehension model

(QANet) for re-ranking the candidate answers based on their fitness for the given question and then sampling the pool of responses with a probability distribution proportional to the scores obtained by each of them (to increase variability). The models integrated by chatbots nowadays include sequence-to-sequence architectures, the Generative Pretraining Transformer (GPT-3), DialoGPT, and the Text-to-Text Transfer Transformer (T5), each of them being assessed in open-domain conversational tasks or subjective evaluations.

Manual evaluations of chatbots consider employing a pool of human assessors who interact with the agent and rate the quality of various conversational aspects using defined metrics and scales. A more efficient manner employs automated evaluations at a lower level of dialogue using metrics such as the F1 score (for measuring accuracy), perplexity (relating to the probability of good prediction of the model), BLEU, and METEOR (originating from the field of machine translation) or even deep methods such as adversarial networks for measuring the distinguishability between the generated and human-like responses. Although impressive models in terms of performance were recently developed, there are still open challenges in terms of naturalness, relevancy, variety of responses, dialogue coherence, knowledge storage, and reasoning. Moreover, there are still no well-established techniques for programmatically evaluating the performances of the chatbots in a comprehensive manner similar to how a human would. It is also worth mentioning the ethical norms that intervene in these systems, such as privacy and data protection, the possibility of offensive or harmful messages, or gender or racial biases generated by the content of the datasets used during model training [17]. These problems are relevant for the practical use of the agents in different scopes, but they add up to the technical complexity and large training datasets required by such chatbots to offer human-like contextualized responses.

A set of natural language processing models and a dialogue state manager are required as a base to build a conversational agent, followed by adding training data, response templates, and eventually producer or consumer service integrations. A toolkit that can ease this development process is RASA [18], which supports various core functionalities, namely the setup of a natural language understanding (NLU) processing pipeline (either sequential or as a graph, starting from its newer 3.0 version), organizing the dataset, training and evaluating the models, and finally binding the agent to various communication channels. RASA also includes configurable models for intent detection, entity extraction, and response or action selection. Furthermore, the support for slots and forms can be leveraged to account for the conversation history during user interactions, coupled with the tracker instance that maintains the state of the dialogue. Another framework to mention in the field of conversational agents is Wit.ai [19], which allows developers to easily build an agent through a step-by-step process and a UI dashboard that abstracts all programming, training, and low-level configurations. In addition, it offers built-in intent, entities, and traits that one can use to add basic assistant abilities to his agent. In this paper, we employed the RASA framework due to the ease of integration of custom language models, spaCy processing pipelines, and knowledge graphs, all of which are presented in the following paragraphs.

A conversational agent has to first determine the intent of the user (i.e., what information is needed or what task he or she wants to be performed by the agent) in order to respond to his or her request. The current intent classification solutions include recurrent neural networks [20], encoder-decoder networks [21], or more recently transformers [22]. One of the state-of-the-art solutions that we also rely on in this paper is DIET [23], which proposes the use of a two-layer transformer for performing the joint task of intent classification and named entity recognition (NER). Both sparse features (based on the constituent words or n-grams of the sentence) and dense features (extracted from pretrained word embeddings) can be fed as input to the transformer. A dimensionality transformation using a feedforward layer is performed in the case of the sparse features to bring them to the same dimension as the dense features, and they are combined afterward through another feedforward layer. The objective of the loss function for intent classification is to maximize the similarity between the predicted intent and the ground truth while minimizing the

similarity between the predicted intent and the negative intent labels. The NER task relies on a conditional random field [24] layer that takes the output of the transformer and tags each sentence token with an entity label or an empty label. Given that the two tasks use the same base features and share a fair part of the network architecture, the model combines both losses for the two subtasks to execute a single round of training that determines the network weights to conform to both objectives simultaneously. This method provides an efficient way of solving both tasks at the same time and also provides better results for the NER due to the transfer of information from the intent classification task.

Along with the main model trained on the application-specific dataset (which in many cases has a relatively reduced size), intent detection can also leverage pretrained models. In this paper, we explore the use of BERT [25], a pretrained transformer-based language model which successfully surpassed the state-of-the-art results in various NLP tasks including question answering, language understanding, and inference. BERT is configured for different sizes by choosing the number of encoders and bidirectional self-attention heads, an ability that supports its training on various corpus volumes. BERT provides contextualized embeddings because of its bidirectional training, token segmentation, and positional embeddings. Another pretrained processing pipeline that we considered was SpaCy [26], which offers components for multiple NLP tasks such as tokenization, part-of-speech tagging, dependency parsing, and named-entity recognition and also provides lower-level embeddings in the form of word vectors. SpaCy also integrates well into the RASA framework, where it can be introduced through built-in components for tokenization and feature extraction (based on word embeddings or lexical information).

While the intent classification task requires good accuracy for the agent to take the correct action in response to the user request, the knowledge representation component should offer a structured but flexible method to persist facts and other more complex information that may come from either the agent developer or from the user him or herself (in the case where the agent learns from the user). NoSQL databases come with a lot of new benefits compared with the SQL alternatives [27], and hence, we chose the former for our current implementation. Aside from their high scalability, ability to run in distributed environments, and ease of development, NoSQL databases provide a lot of flexibility, as they do not require defining a fixed and strict schema for all stored data but instead allow much more variation in data types and relationships between different pieces of information. Consequently, they represent a much better alternative for the case of knowledge representation and especially for natural language knowledge, where facts are, in general, not homogeneous, and the introduction of new types of facts would require different schemas and implementations for storing and finding information.

Among the NoSQL database types, graph databases have been proven to be most suitable type for natural knowledge representation (especially in a conversational agent environment) because of the match between their structure and the way the tokens or the semantic entities of a sentence and the dependencies between them are usually represented (parse trees or graphs) when performing a syntactic or semantic analysis of a statement [28]. In terms of data modeling in graph databases, it is worth mentioning the RDF triple datastore, which was standardized by W3C as a model for interchanging data in the Semantic Web [29]. RDF provides a strict but powerful structure through its URI which is used to uniquely identify the node entities and relationships between them and the triples format for defining the connections between a subject and an object using predicates.

On a higher level of abstraction, the Semantic Web is designed to leverage the automated processing of information through links between entities of interest. Ontologies are tightly linked to the Semantic Web [30] and introduce powerful reasoning abilities, starting from explicitly known definitions of classes, individuals, and properties as links between entities. As such, these data representation models can be used in the context of tasks such as information retrieval or question answering, which are essential for the envisioned conversational agent.

3. Method

The natural language understanding of user utterances consists of splitting the sentence into tokens, composing the set of features (extracted at the sentence, token, or n-gram level), detecting the intent and entities, and applying additional processing techniques such as syntactic parsing. Finally, based on the detected intent and filled slots (entities), a response utterance or action is selected as feedback to the user input.

3.1. Microworld Definition

The intent detection task relies on a two-stage classification for multi-domain adaptation, where the former places the sentence in one of the defined microworlds while the latter is responsible for identifying the specific intent of the sentence within that informational context. A similar approach was also explored by Smith, Williamson, Shuster, Weston, and Boureau [31]. The rationale behind this multi-level classification is to enable the definition of specialized classifiers that can be trained and fine-tuned individually for each subtask. Only a fixed set of classes (microworlds) is available for the classification between domains, as this can lead to better classification confidence and fewer possible confusion types. Furthermore, the second-level classifiers are aware only of the intents belonging to their microworld. Thus, an easier distinction between user intents is possible.

An additional benefit of this modular architecture is the possibility of easily adding or removing a microworld (plug in or out). This can be accomplished by simply bringing the domain-specific intent classification model into the pipeline and retraining the first-level classifier only. In RASA, this implies creating a custom processing component by overriding the training and prediction methods of the base classifier. A set of custom actions for performing external operations or knowledge base queries can be bundled together with the microworld dataset for more practical use cases, together making up a self-contained conversational module.

In our implementation, the intent detection task leverages the DIET classifier provided by the RASA framework. The first-level classifier receives as training examples the entire dataset, where the labels are adjusted to reflect only the microworld name. During prediction, its output (i.e., the domain) is passed forward to be used by the following components in the pipeline. Based on the resulting domain of the sentence, the second-level classifier is picked accordingly, so the predicted intent always belongs to the initially assumed domain. The final confidence of the prediction can be computed as the product of the two classification confidences.

As a practical implementation, three microworlds were put together to build a multi-domain agent that could act as a personal assistant as follows.

3.1.1. Microworld for University Guidance

This microworld targets providing a quick and easy way to access information regarding the University Politehnica of Bucharest, more specifically the information about professors (e.g., their offices), guidance for reaching a specific classroom, or the classroom or time interval for a specific activity (course, seminar, or laboratory) of a student's group [32].

There are three main types of RDF subjects involved, namely professors, rooms, and activities, each of them with properties that can be presented by the agent to the user:

- Professors: name, description, teaching degree, and office;
- Classrooms: ID and direction (guidance description);
- Activities: ID, name, student series or groups, room (link to a classroom node), teacher name, type (course, seminar, or laboratory), and time slot (workday, time, and duration).

The intents defined include getting the room ID of a professor's office, getting indications for reaching a room, and finding the time interval and room for activities in different situations: the user specifies only the subject name, the subject, and the activity type or also the student group. In case any of the required information was not specified in the

initial request, the user would be asked for it in the next replies until all the RASA slots were filled.

3.1.2. Microworld for Memory Assistance

The memory assistance abilities of the agent consist of extracting factoid information from user statements (such as the moment of an event, the placement of an object, or different properties of some entities expressed through a self-contained text value), as well as reminding the agent back to the user when requested through a question [2]. For this task, a classifier trained on a set of intents specific to storing and retrieving each type of information was used. On the knowledge base side, each type of fact uses a specific predicate for the RDF triples (e.g., location, time point, time start, time end, time range, duration, and entity attribute). Multiple properties (as simple nodes or more complex trees) are linked to the main node, which is the action definition, to store more complex structures such as the actions, which may have multiple syntactic elements attached such as direct or indirect objects or adverbials of the place or time.

3.1.3. Generic Microworld

This microworld encompasses all general and unclassified intents the user may express, such as greetings, thanks, confirmations, and requests for help in the interaction with the agent (asking for the agent's abilities). Its role is to take over any sentence that does not fit into the other, more specific microworlds. This is the simplest microworld, and on the output side, it consists only of predefined responses (eventually with multiple reformulations for the same message) without custom actions. In the future, this microworld might be used to handle general knowledge questions using an external data source, such as Wikidata [33] or DBpedia [34], which can be queried through SPARQL [35]) as well. This further emphasizes the adequacy of using knowledge graphs for presenting agent data.

3.2. Language Models

In its simplest form, a RASA agent relies only on the content of the developer-provided dataset for training the NLU pipeline, which can work for simple use cases with well-defined intents that can be easily distinguished (e.g., by specific keywords), considering that a sufficient variety of examples is available. However, for complex applications where the dialogue can span across a larger domain of knowledge, and the agent needs to deeply understand the semantic meaning of the words within the language, pretrained embeddings are more suitable.

For the Romanian conversational agent in the discussion, three different pretrained language models were evaluated within the task of intent classification. First, a spaCy model [36] based on the RASA framework recommendations and pretrained on the Romanian Universal Dependencies treebank was explored [37,38]. It offers dense word vectors, as well as other models such as part-of-speech tagging or dependency parsing, which can be used to extract lexical and syntactic features. Second, an uncased large BERT model pretrained for Romanian (341M weights) [39] was introduced using the Language Model Featurizer RASA component. Finally, a distilled version of BERT [40], obtained from the base version of the previous model and another BERT-cased model, was tested similarly. The results section presents a practical comparison between the three embeddings.

3.3. Syntactic Features

Reiterating the remark that, currently, most conversational agents are oriented toward a specific context and practical usage, we can leverage not only named entities or keywords specific to the context but also more abstract syntactic information that can indicate one semantic construction or another. Concretely, we propose the use of syntactic questions that can be raised for each word of a sentence to get to its syntactic function as additional features and improve intent detection. Figure 1 illustrates the entire flow, starting from

a sentence and ending up with the intent classifier, which uses the generated features as input.

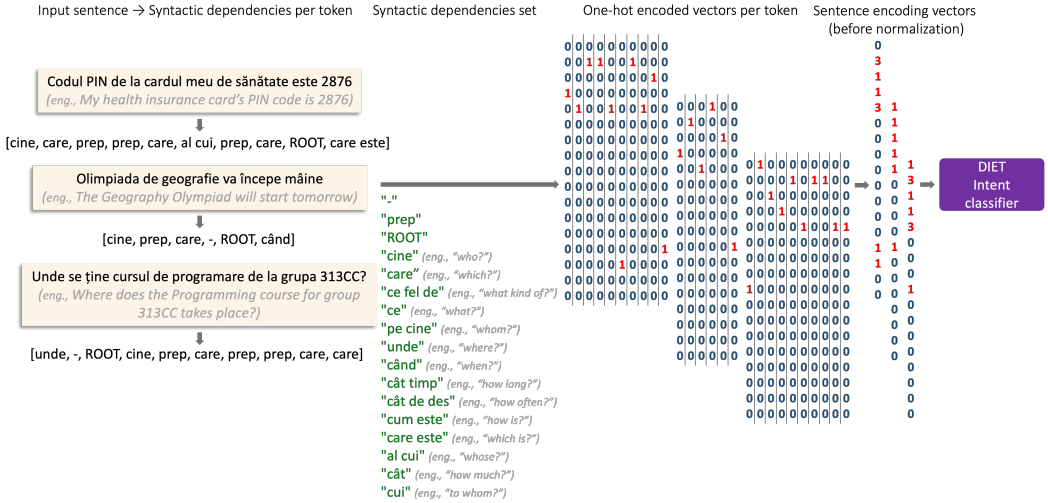

Figure 1. Flow for generating the syntactic features.

The syntactic parsing model previously described by Boroghina et al. [2] was leveraged in the first place to generate the syntactic dependencies used for intent detection in the RASA agent. The role of the parser is to consider the syntactic function of words and to extract the directed binary grammatical relations between the heads and dependents. The following labels are used in the parsing process, in accordance with Romanian grammar [41]:

- "Cine?" (Eng., "who?"): defines the subject of the action;
- "ROOT": label marking the verb;
- "Pe cine?", "ce?" (Eng., "whom?", "what?"): identify a direct object;
- "Care?", "ce fel de?" (Eng., "which?", "what kind of?"): identify the attributes of a noun or a substitute of a noun (creating together a noun phrase, or a grammatical structure that performs syntactically similar to a single noun);
- 'Unde?" (Eng., "where?"): identifies a location complement that can define a source or destination location (depending on the prepending prepositions);
- "Când?", "cât timp?", "cât de des?" (Eng., "when?", "how long?", "how often?"): identify a temporal complement defining a point in time, a duration, or a frequency, respectively;
- "Care este?", "cum este?" (Eng., "which is?", "how is?"): describe tokens representing the predicative of a nominal predicate where, semantically, they express the value and qualification of the subject, respectively;
- "Al cui?" (Eng., "whose?"): marks the ownership of the entity;
- "Cui?" (Eng., "to whom?"): identifies an indirect object;
- "Prep": label marking a preposition (no syntactic role, but necessary for linking the tokens in structures such as noun phrases);
- "Cât?" (Eng., "how much?"): describes a quantity of the entity that it determines, and it is expressed in general by a numeral or a quantitative adjective such as "mult" (Eng., "much") or "puțin" (Eng., "little").

All other head-token relations that do not fall under any of the previously mentioned classes are marked with the"-" (undefined) label. Figure 2 exemplifies the syntactic parsing annotation for two simple sentences.

```
{
  // "sentence": "The Communication Protocols course test was on Thursday"
  "sentence": "Testul (0) de (1) curs (2) la (3) Protocoale (4) de (5) Comunicații (6) este (7) joi (8)",
  "deps": [
    /* 7 -> 0 */ "cine",
    /* 2 -> 1 */ "prep",
    /* 0 -> 2 */ "care",
    /* 4 -> 3 */ "prep",
    /* 0 -> 4 */ "care",
    /* 6 -> 5 */ "prep",
    /* 4 -> 6 */ "ce fel de",
    /* 7 -> 7 */ "ROOT",
    /* 7 -> 8 */ "când"
  ]
},
{
  // "sentence": "Where is the swimming pool?"
  "sentence": "Unde (0) se (1) află (2) bazinul (3) de (4) înot (5)?",
  "deps": [
    /* 2 -> 0 */ "unde",
    /* 2 -> 1 */ "-",
    /* 2 -> 2 */ "ROOT",
    /* 2 -> 3 */ "cine",
    /* 5 -> 4 */ "prep",
    /* 3 -> 5 */ "care"
  ]
},
```

Figure 2. Syntactic parsing with annotated examples (token IDs are the corresponding numbers in parenthesis).

The model leverages the spaCy dependency parsing model by retraining it using a syntactic dependencies dataset to identify the new type of relations (between a word and the token that it is determining from a syntactical point of view) and tag these dependencies with the syntactic questions described above. However, to avoid getting inconclusive results for the intent detection performance gain due to the imperfect predictions of the syntactic model, manual corrections of the entire NLU dataset were performed in terms of the extracted relevant syntactic dependencies.

By putting together all the syntactic questions from the sentence, the agent can get a hint about the type of information requested or presented by the user, whether it is a request for an operation depending on locations (e.g., planning travel, making a booking, or finding details about something related to a location), timestamps (e.g., event-related statements), attributes (e.g., descriptions), or facts about a subject (e.g., definitions).

The syntactic questions identified for each word have to be processed into a set of numeric values that can be fed as input to the DIET classifier. A dense featurizer was developed to precede the intent classifier in the NLU pipeline. Considering the categorical nature of the syntactic questions, an approach to generate dense feature vectors is to build one-hot encoded vectors with the same length as the list of syntactic questions, with a value of one for the position of the syntactic dependency corresponding to the token and zero for all other positions. Therefore, the dense syntactic featurizer takes a previously tokenized sentence and builds one-hot encoded word vectors for each token. The word vector corresponding to the sentence is then computed through a mean pooling operation (i.e., a normalized average of the token vectors in the multidimensional space defined by the set of syntactic dependencies). As can be noted, this approach takes into consideration only the number of occurrences for each syntactic question, disregarding their positions within the sentence.

3.4. Knowledge Representation and Retrieval

All facts used to respond to user questions are stored in a single knowledge graph based on RDF triples. This means that for any subject node (entity), we can attach a

property through a predicate connection that goes to an object node. In contrast to the labeled property graph model, where we can add an arbitrary number of key-value pairs to any node, in RDF, all information is modeled through separate nodes and arcs. Thus, more complex structures are composed by adding multiple predicate-object extensions to a subject. Most of the currently defined properties are simple literal nodes (containing a string or numeric values), but for increased understandability of the data and more inference capabilities, more complex object nodes may be used (e.g., for time slots by separating the day, hour, and duration or for teachers by linking the teacher's node from the graph). A listing of the possible classes (i.e., subjects) and associated properties (i.e., predicates) that intervene in the proposed microworlds is defined in Table 1. The list is not exhaustive and can be extended or integrated with other data types from the current domains (e.g., the memory assistant can theoretically introduce any kind of data that may be expressed through single-predicate statements) or with new domains to be considered within the conversational agent.

Table 1. Knowledge base classes and properties.

Microworld	Class	Properties
University Guidance	Professor	Name, academic position, description, office
	Room	ID, location indications
	Activity	Type, name, ID, student groups, classroom, time slot
Memory Assistance	Entity	Location, description, specifier
	Event	Timestamp
	Property of entity	Value
	Subject	Owned entity

The implementation relies on the GraphDB DBMS [42], which provides native support for storing semantic graphs (RDF triplestores). Although no object graph mapping for translating between a Python representation of the triples and the RDF format exists, GraphDB supports the RDF4J REST API specification [43], which can be used to send SPARQL queries and updates to the database. Predefined knowledge (e.g., university information) is described through suggestively named subject-predicate-object triples (see Figure 3) in Turtle files, which can be loaded into the database using the LoadRDF tool [44]. Dynamic information (e.g., user-provided facts) is inserted into the graph during conversations. Moreover, given that the number of nodes implied in the matching might be arbitrary depending on the type of information involved, an identification based on UUIDs is implemented. Concretely, each subject and object of a triple representing intermediary nodes are defined through UUIDs generated on the fly. On store operations, the unicity of the UUIDs ensures the nodes that are created in the knowledge graph (e.g., nodes that represent instances of a class) do not clash with other existing nodes.

```
:John_Doe
    a            :Professor ;
    :coord       "Assistant Professor"@en, "Șef de lucrări"@ro ;
    :description "Assistant lecturer at the computer science department"@en ;
    :name        "John Doe" ;
    :office      "ROOM01" .
```

Figure 3. Professor's information in Turtle format (RDF).

An intuitive depiction of the database facts is represented by the visual graph that links all the subject and property nodes through directed arcs (predicates) as seen in Figure 4, which can be generated from the GraphDB workbench.

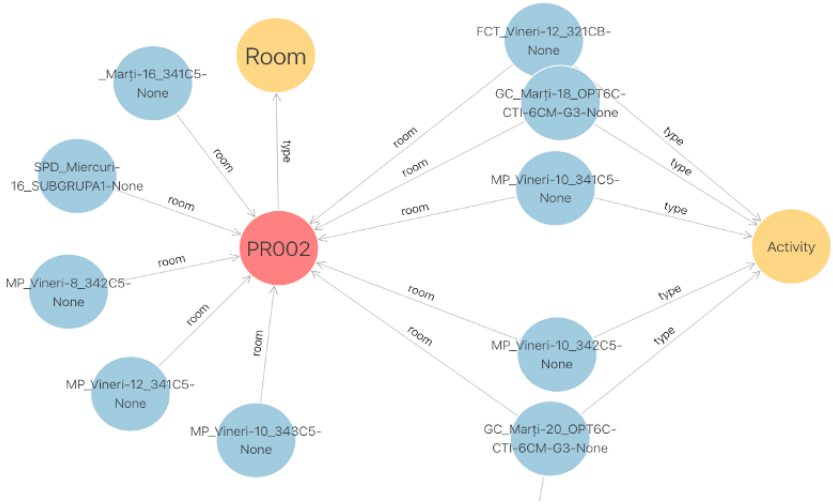

Figure 4. GraphDB's knowledge graph visualization.

Database queries are performed in SPARQL by concatenating one or more RDF triple matching instructions as follows:

$$? < subj > a :< class >; \ldots :< pred_i >:< obj_i >; \ldots$$

The above query matches a subject node belonging to a given class and having some other attributes. Figure 5 depicts on the left side a SPARQL query example for extracting the timetable activities (university guidance) from the database, where the ID, name, type, and room are direct properties of the activity, the teacher is an optional property, and the time slot is represented by other intermediary nodes that have attributes such as the day, time, and duration. The results returned from the database (see the right side of the figure) are structured as a list of (key and value) pairs, where the key is the name of the node to be matched, used in the select clause of the query, and the value contains the value matched to that specific key. Each match generates a separate entry in the results list, from which the user picks the desired one if the dialogue is to be continued based on the response.

```
select ?id ?type ?room ?teacher ?group {
    ?activity a :Activity ;
              :id ?id;
              :name ?name;
              :type ?type;
              :room ?room.
    ?activity :teacher ?teacher .
    ?activity :groups [
        :group ?group;
        :semigroup ?semigroup
    ].
    ?activity :timeSlot [
        :day ?day;
        :time ?time;
        :duration ?duration;
    ].
}
```

	id	type	room	teacher	group
1	"AFNEN"	"course"	:AN210	"NICOLETA IGNAT"	"OPT6C-CTI-6CM-G3"
2	"PM"	"course"	:PR001	"D. TUDOSE"	"333CA"
3	"TS"	"course"	:PR001	"V. CALOFIR"	"321CB"
4	"BD"	"course"	:PR001	"AL. BOICEA"	"333CA"
5	"SSI"	"p"	:PR605	"C. DOBRE"	"OPT6C-CTI-6CM-SSA"
6	"MFDA"	"course"	:PR408	"ECATERINA OLTEAN"	"SEMIGRUPA3"
7	"TSW"	"course"	:EG301	"B. DUMITRESCU"	"OPT6C-CTI-6CM-G3"

Figure 5. Query and response for timetable activity extraction (SPARQL).

The capability to decompose and store dynamic data is illustrated in Figure 6, where two sentences with one common entity—"portofel" (Eng., "wallet")—are stated to define location information. Given that there can be more than one instance of the base class of

an entity, each one having different specifiers or owners, an additional instance node is needed in the representation of articulated noun phrases to link all the components and enable single-point access to that particular instance (e.g., "cardul de credit" (Eng., "the credit card")). The structure generated in the graph can be seen in Figure 7. One simple inference that can be applied in this case would be using chained SPARQL triples to find the actual (final) location (i.e., "geanta albastră") of the subject (i.e., "cardul de credit").

```
"Cardul de credit este în portofel"
(eng., "The credit card is in the wallet")

INSERT DATA {
:8aea5d1e43b3440e893fb52d0885774a
    a       :Class ;
    :value  :card ;
    :pre    "" .

:36bc1bb232c14e6ab4546e6082c14bdf
    a      :Instance ;
    :IS_A  :8aea5d1e43b3440e893fb52d0885774a .

:d7d5a7a1da8a442281fb037d4c4603d7
    a       :Class ;
    :value  :credit ;
    :pre    "de" .

:36bc1bb232c14e6ab4546e6082c14bdf
    :SPEC  :d7d5a7a1da8a442281fb037d4c4603d7 .

:36bc1bb232c14e6ab4546e6082c14bdf
    :value "cardul de credit" .

:d0f1bcb057704f53a185558c395aa602
    a       :Class ;
    :value  :portofel ;
    :pre    "în" .

:36bc1bb232c14e6ab4546e6082c14bdf
    :LOC  :d0f1bcb057704f53a185558c395aa602 .
} ;
```

```
"Portofelul este în geanta albastră"
(eng., "The wallet is in the blue bag")

INSERT DATA {
:09198bc9a30046a283ccdfa493e55eb2
    a       :Class ;
    :value  :portofel ;
    :pre    "" .

:d1cbd2240bdf4ecca869f17aaec34cb9
    a       :Class ;
    :value  :geanta ;
    :pre    "în" .

:e6aeac2000164a5a816e8a05892489a0
    a      :Instance ;
    :IS_A  :d1cbd2240bdf4ecca869f17aaec34cb9 .

:dc3a4f012c6049089c1232a0937da2dc
    a       :Class ;
    :value  :albastră ;
    :pre    "" .

:e6aeac2000164a5a816e8a05892489a0
    :SPEC  :dc3a4f012c6049089c1232a0937da2dc .

:e6aeac2000164a5a816e8a05892489a0
    :value "în geanta albastră" .

:09198bc9a30046a283ccdfa493e55eb2
    :LOC  :e6aeac2000164a5a816e8a05892489a0 .
} ;
```

Figure 6. SPARQL queries for inserting dynamic information into the knowledge graph.

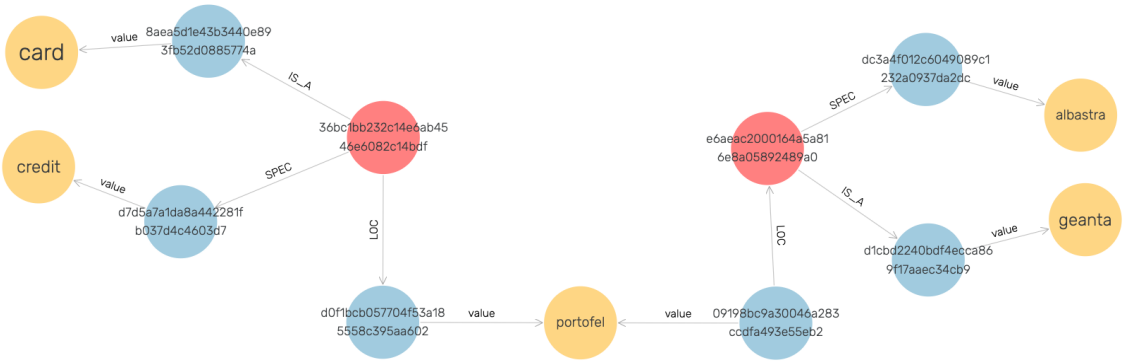

Figure 7. Knowledge sub-graph illustrating a set of dynamically inserted data.

4. Results

Compared with the initial experiments of Boroghina et al. [2], due to the introduction of the concept of multiple microworlds within the same conversational agent instance, the dataset used for training the natural language understanding pipeline was extended and refactored to account for the domain of knowledge, as well as for the specific intent the examples belong to. We also made the code open source at https://github.com/readerbench/conversational-agent (accessed on 2 May 2022). Moreover, the configuration of the pipeline was updated to accommodate the two-stage intent classification. Thus, we provide an evaluation of the new configuration of the agent. From the set of approximately 600 total examples that were made publicly available at https:

//huggingface.co/datasets/readerbench/ConversationalAgent-Ro (accessed on 2 May 2022), we generated an 80/20 train/test split, with a proportional number of support examples in the train versus test sets for each intent.

The first evaluation revealed the performance of the multi-domain pipeline concerning the intent classification task. Figure 8 shows the confusion matrix obtained by aggregating the misclassifications produced by the agent in the pool of 21 defined intents from the test examples.

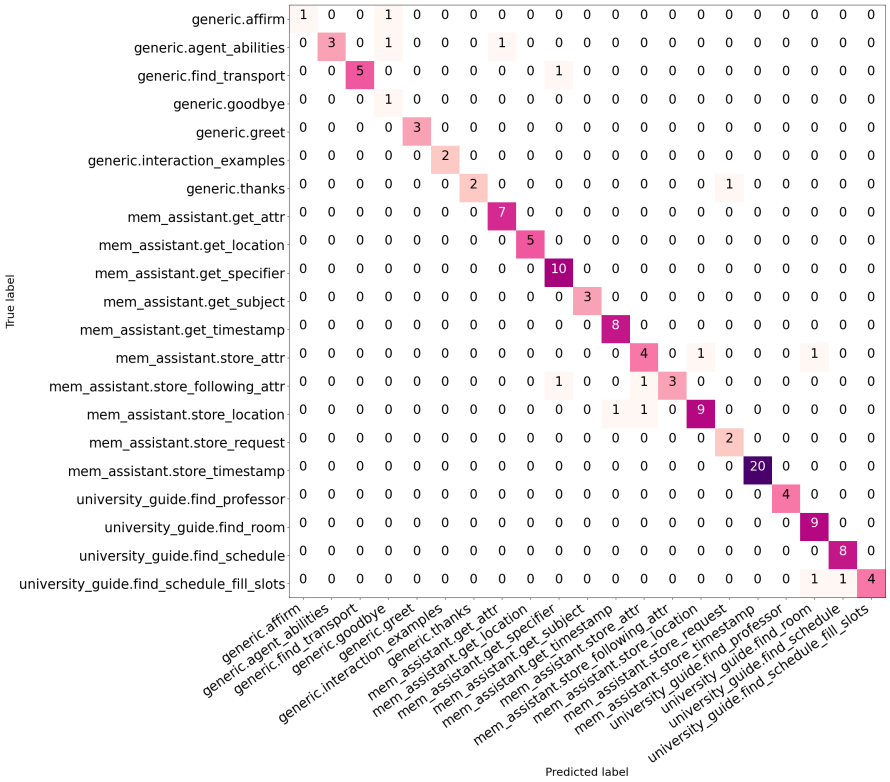

Figure 8. Multi-microworld intent classification's confusion matrix.

Next, we evaluated the two explored pipeline improvements, pretrained language models, and syntactic features, both individually and together. Table 2 provides a numeric comparison (i.e., mean and standard deviation performance scores) of the different pipeline configurations after 10 independent train and test runs. The first row is the baseline model in which no improvements were made to the default Rasa pipeline, whereas the second row highlights the advantage of using syntactic dependencies as features in the DIET intent classifier. The next three rows evaluate the performances of the three considered language models. Finally, the last row presents the gains obtained by combining the RoBERT large model with the syntactic features, thus resulting in the configuration with the best results. We observe that the syntactic features increased the F1 score by more than 2% for both scenarios (i.e., no language model and with RoBERT large), whereas considering language models for Romanian increased the F1 score even further up to 82.6%.

Table 2. Comparison of NLU pipeline variations for intent classification (bold denotes the best performance).

Language Model	Syntactic Features	Precision M (SD)	Recall M (SD)	F1 Score M (SD)
None	No	79.2 (2.5)	76.7 (2.6)	75.9 (2.9)
None	Yes	80.7 (3.3)	78.5 (3.1)	77.9 (3.2)
SpaCy Ro	No	80.5 (2.4)	77.8 (2.8)	77.1 (3.0)
DistilMulti-BERT-base-ro	No	78.2 (2.7)	75.9 (2.4)	75.2 (2.6)
RoBERT large	No	82.4 (2.3)	81.0 (2.5)	80.1 (2.9)
RoBERT large	Yes	**85.2 (3.0)**	**83.2 (3.1)**	**82.6 (3.1)**

Figure 9 depicts the performance gains obtained in several runs when trying to leverage different proportions from the training dataset from 60% to 100%. We performed six train and test runs for each training subset.

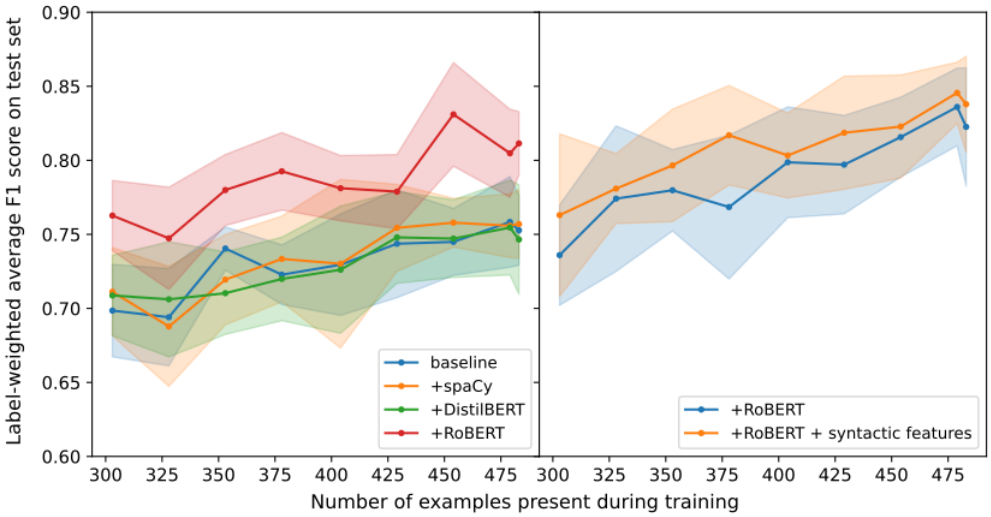

Figure 9. Comparison of NLU pipeline variations.

5. Discussion

Concerning per-intent classification mistakes, a first remark is that the confusions performed by the agent were relatively sparse and reduced considering the high number of defined intents. Additionally, it can be noticed that most confusions were intra-microworld, meaning that the individual classifiers may require further improvements, while the top-level classifier provided correct detection of the microworlds. These results come in support of using the two-stage pipeline when considering multiple domains by splitting the intent classification effort into two easier classification subtasks. In Table 3, the confusions produced by the intent classification component after a train and test run are presented and discussed to provide insights regarding particular situations that may affect detection.

Table 3. Error analysis for intent classification.

Sentence	Real Intent	Predicted Intent	Confi-Dence	Discussion
Ok	generic. affirm	generic. goodbye	90.6	Short sentence (harder for count vectors), foreign words (harder for language model)
Mi-ai fost de mare ajutor. (Eng.: You have been really helpful.)	generic. thanks	mem_assistant. store_request	64.4	Insufficient generic microworld support in the data set
Care sunt lucrurile pe care știi să le faci? (Eng.: What are the things that you know how to do?)	generic. agent_abilities	mem_assistant. get_attr	99.9	Phrase "care sunt" (Eng.: what are) similar to the predicted intent
Help	generic. agent_abilities	generic. goodbye	45.1	Short sentence (harder for count vectors), foreign words (harder for language model)
Hotelul la care ne-am cazat anul trecut la mare e acesta. (Eng.: The hotel where we stayed last year at the seaside is the following.)	mem_assistant. store_following_attr	mem_assistant. get_specifier	99.9	Phrase "la care" (Eng.: where) specific to the predicted intent
Programul de la notarul de la Universitate este acesta. (Eng.: The program from the notary at the University is the following.)	mem_assistant. store_following_attr	mem_assistant. store_attr	60.0	Examples of the two intents have similar prefixes
Sunt în grupa 232. (Eng.: I'm in group 232.)	university_guide. find_schedule_fill_slots	university_guide. find_room	77.3	Sequence (232) interpreted as room ID
Cursul de Programare Orientata pe Obiecte (Eng.: Object-Oriented Programming Course)	university_guide. find_schedule_fill_slots	university_guide. find_schedule	99.9	Some "find_schedule_fill_slots" examples are subsequences of "find_schedule"-specific sentences
Cum pot ajunge rapid la laserul de la Măgurele? (Eng.: How can I quickly get to the Magurele laser?)	generic. find_transport	mem_assistant. get_specifier	34.1	No obvious justification for confusion, but the low confidence indicates possible wrong detection
Sala laboratorului de PP este EG321. (Eng.: The PP laboratory room is EG321)	mem_assistant. store_attr	university_guide. find_room	94.8	Entities specific to the university guidance microworld (activity type: laboratory, classroom: "EG321")
Suprafața apartamentului de la București este de 58 mp. (Eng.: The surface of the apartment in Bucharest is 58 sqm.)	mem_assistant. store_attr	mem_assistant. store_location	99.9	Sentence contains a location, although as an attribute and not a complement
Am mers cu mașina până la Iași. (Eng.: We drove by car to Iasi.)	mem_assistant. store_location	mem_assistant. get_timestamp	42.8	No obvious justification for confusion, but the low confidence indicates possible wrong detection
Chitara mea este de la Andrei. (Eng.: My guitar is from Andrei.)	mem_assistant. store_location	mem_assistant. store_attr	75.4	Phrase "<subj> este" similar to the predicted intent

One other remark is on the pretrained embeddings side, where the RoBERT model delimits itself as it provides a significant gain in intent classification accuracy. The other

language models do not provide a considerable boost in performance on average compared with the baseline pipeline, which relies mainly on count vectors as features. Furthermore, we can observe the benefits of adding features coming from syntactic questions on top of the RoBERT variant, which gave an increase of approximately 2% concerning the F1 scores. In contrast, the distilled version of BERT seemed to offer worse representations, given the scores that were below a baseline where no language model or syntactic features were used.

An essential dimension is the quality and quantity of the dataset used for training the NLU pipeline that enables intent detection and the extraction of entities (the DIET classifier), as well as the syntactic components of a sentence (the syntactic parser) since these represent the core of the current conversational agent and can seriously impact the user-perceived quality. For instance, intents from different microworlds may appear similar to the agent and might become confused (e.g., finding the office of a professor versus finding the place of an object), which leads to a completely erroneous interpretation of the user input in the wrong microworld.

One limitation of the multi-domain approach presented in this paper is that it allows a finite number of domains to be integrated and cannot support a fully open domain mode because each request-reply exchange is interpreted in its corresponding microworld. As such, cross-domain inferences are more complex. This means that the proposed agent is more suitable for task-oriented applications rather than a chatbot, which implies a longer cohesive dialogue.

Regarding knowledge representation and retrieval, the proposed approach was validated using a set of real-world data extracted and curated from the timetables and various sites of the faculty (for schedule information and data about professors), as well as by discussion with the agent for introducing dynamic, unpredictable data. Information retrieval can be performed by matching noun phrases and extracting the properties that relate to them or eventually by traversing the knowledge graph and building longer inference chains based on rules (e.g., merging same-type adjacent relations such as location or ownership). Future improvements in this area may include more intelligent methods for combining and extracting facts from the knowledge graph by making machine learning-based inferences on the accumulated data.

Overall, we introduced a conversational agent for the Romanian language designed as a personal assistant that becomes domain-aware and learns new information dynamically from user conversations in multiple microworlds by means of a knowledge graph. We ensured high capabilities for natural language understanding (F1 score = 82.6) using the proposed pipeline that takes into account RoBERT-contextualized embeddings alongside syntactic features, which surpassed by a considerable margin the baseline model. In addition, we argue that the proposed RDF knowledge representation provides flexibility to store and retrieve natural language entities, values, and factoid relations between them in the context of each microworld.

6. Conclusions

The current paper comes as an extension of our previous work [2] in the field of conversational agents integrated with graph knowledge stores. Here, we added new advances to the standard NLU pipeline that one can build by using the RASA framework, accompanied by practical evaluations. We also proposed a more representative data modeling approach within a knowledge base based on RDF. Moreover, we tackled the extended use of agents in a multi-domain fashion by including a few knowledge contexts which can form a minimal personal assistant. Lastly, we made our dataset and code open source on GitHub [14] and Huggingface [15], respectively.

An area of possible enhancement regarding the capabilities of the conversational agent is dialogue management. Although RASA provides support and features for managing the flow of a conversation or directing a conversation toward filling some information slots required by the agent to solve user requests, the current intents do not involve or expect such longer dialogues but instead simple request-response message exchanges. The agent

could behave more intelligently and human-like, aside from providing hard-structured answers by defining more complex stories that can account for the history of the discussion and provide contextualized replies.

The validation of the model can also be enhanced by human interactions. Users can help the development of the model by providing feedback on the responses of the bot in both normal and stress test scenarios. As such, we will investigate the extent to which the dynamic increase of the knowledge graph impacts the performance of the chatbot.

Finally, building a version of the agent in a more widespread language such as English would increase the usefulness and possible applications of the agent in real-life situations. This conversion would imply creating a dataset for intent detection, namely one for syntactic parsing of the sentences, and training new models. Possible approaches would be a simple translation of the current dataset examples or a semi-automated generation by crawling examples from various sources, given that more corpora are already available for English.

Author Contributions: Conceptualization, G.B., D.G.C. and M.D.; methodology, G.B. and M.D.; software, G.B. and D.G.C.; validation, G.B., D.G.C. and M.D.; formal analysis, G.B. and M.D.; investigation, G.B., D.G.C. and M.D.; resources, G.B. and D.G.C.; data curation, G.B.; writing—original draft preparation, G.B.; writing—review and editing, D.G.C. and M.D.; visualization, G.B.; supervision, M.D.; project administration, M.D.; funding acquisition, M.D. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by a grant from the Ministry of Research, Innovation and Digitization (CNCS/CCCDI-UEFISCDI, project number PN-III-P2-2.1-SOL-2021-2-0223) within PNCDI III.

Data Availability Statement: The data corresponding to all three considered microworlds are openly available in HuggingFace at https://huggingface.co/datasets/readerbench/ConversationalAgent-Ro (accessed on 2 May 2022).

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

Abbreviations

The following abbreviations are used in this manuscript:

BERT	Bidirectional encoder representations from transformers
DBMS	Database management system
DIET	Dual Intent and Entity Transformer
NER	Named-entity recognition
NLP	Natural language processing
NLU	Natural language understanding
RDF	Resource Description Framework
SPARQL	SPARQL Protocol and RDF query language
UUID	Universally unique identifier

References

1. Montenegro, J.L.Z.; da Costa, C.A.; da Rosa Righi, R. Survey of conversational agents in health. *Expert Syst. Appl.* **2019**, *129*, 56–67. [CrossRef]
2. Boroghina, G.; Corlatescu, D.G.; Dascalu, M. Conversational Agent in Romanian for Storing User Information in a Knowledge Graph. In Proceedings of the 17th International Conference on Human-Computer Interaction, RoCHI 2020, Sibiu (Virtual), Romania, 22–23 October 2020; pp. 95–102. [CrossRef]
3. Adewumi, T.; Liwicki, F.; Liwicki, M. State-of-the-Art in Open-Domain Conversational AI: A Survey. *Information* **2022**, *13*, 298. [CrossRef]
4. Caldarini, G.; Jaf, S.; McGarry, K. A Literature Survey of Recent Advances in Chatbots. *Information* **2022**, *13*, 41. [CrossRef]
5. Beckett, D.; McBride, B. RDF/XML syntax specification (revised). *W3C Recomm.* **2004**, *10*, 1–56.
6. Ait-Mlouk, A.; Jiang, L. KBot: A Knowledge graph based chatBot for natural language understanding over linked data. *IEEE Access* **2020**, *8*, 149220–149230. [CrossRef]
7. Gerontas, A.; Zeginis, D.; Promikyridis, R.; Androš, M.; Tambouris, E.; Cipan, V.; Tarabanis, K. Enhancing Core Public Service Vocabulary to Enable Public Service Personalization. *Information* **2022**, *13*, 225. [CrossRef]

8. Promikyridis, R.; Tambouris, E. Using Knowledge Graphs to provide public service information. In Proceedings of the DG. O 2022: The 23rd Annual International Conference on Digital Government Research, Virtual, 15–17 June 2022; pp. 252–259.
9. Bao, Q.; Ni, L.; Liu, J. HHH: An Online Medical Chatbot System based on Knowledge Graph and Hierarchical Bi-Directional Attention. In Proceedings of the Australasian Computer Science Week, ACSW 2020, Melbourne, VIC, Australia, 3–7 February 2020; pp. 32:1–32:10. [CrossRef]
10. Ni, P.; Okhrati, R.; Guan, S.; Chang, V. Knowledge Graph and Deep Learning-based Text-to-GraphQL Model for Intelligent Medical Consultation Chatbot. *Inf. Syst. Front.* **2022**, 1–20. [CrossRef]
11. Breitfuss, A.; Errou, K.; Kurteva, A.; Fensel, A. Representing emotions with knowledge graphs for movie recommendations. *Future Gener. Comput. Syst.* **2021**, *125*, 715–725. [CrossRef]
12. Varitimiadis, S.; Kotis, K.; Pittou, D.; Konstantakis, G. Graph-Based Conversational AI: Towards a Distributed and Collaborative Multi-Chatbot Approach for Museums. *Appl. Sci.* **2021**, *11*, 9160. [CrossRef]
13. Varitimiadis, S.; Kotis, K.; Spiliotopoulos, D.; Vassilakis, C.; Margaris, D. "Talking" Triples to Museum Chatbots. In *Proceedings of the Culture and Computing—8th International Conference, C&C 2020, Held as Part of the 22nd HCI International Conference, HCII 2020, Copenhagen, Denmark, 19–24 July 2020*; Lecture Notes in Computer Science; Rauterberg, M., Ed.; Springer: Berlin/Heidelberg, Germany, 2020; Volume 12215, pp. 281–299. [CrossRef]
14. Github. Conversational Agent Project. 2022. Available online: https://github.com/readerbench/conversational-agent (accessed on 2 May 2022).
15. HuggingFace. HuggingFace. 2021. Available online: https://huggingface.co/ (accessed on 1 February 2022).
16. Hardalov, M.; Koychev, I.; Nakov, P. Machine Reading Comprehension for Answer Re-Ranking in Customer Support Chatbots. *Information* **2019**, *10*, 82. [CrossRef]
17. da Silva, D.A.; Louro, H.D.B.; Goncalves, G.S.; Marques, J.C.; Dias, L.A.V.; da Cunha, A.M.; Tasinaffo, P.M. Could a Conversational AI Identify Offensive Language? *Information* **2021**, *12*, 418. [CrossRef]
18. Bocklisch, T.; Faulkner, J.; Pawlowski, N.; Nichol, A. Rasa: Open source language understanding and dialogue management. *arXiv* **2017**, arXiv:1712.05181.
19. Wit.ai. Wit.ai. 2022. Available online: https://wit.ai/ (accessed on 1 March 2022).
20. Mikolov, T.; Karafiát, M.; Burget, L.; Černocký, J.; Khudanpur, S. Recurrent neural network based language model. In Proceedings of the Eleventh Annual Conference of the International Speech Communication Association, Makuhari, Japan, 26–30 September 2010.
21. Sutskever, I.; Vinyals, O.; Le, Q.V. Sequence to sequence learning with neural networks. In Proceedings of the Advances in Neural Information Processing Systems, Montreal, QC, Canada, 8–13 December 2014; pp. 3104–3112.
22. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, L.; Polosukhin, I. Attention is all you need. In Proceedings of the Advances in Neural Information Processing Systems, Long Beach, CA, USA, 4–9 December 2017; Volume 30, pp. 5998–6008.
23. Bunk, T.; Varshneya, D.; Vlasov, V.; Nichol, A. DIET: Lightweight Language Understanding for Dialogue Systems. *arXiv* **2020**, arXiv:2004.09936.
24. Lafferty, J.; McCallum, A.; Pereira, F.C. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In Proceedings of the ICML, Williamstown, MA, USA, 8 June–1 July 2001; pp. 282–289.
25. Devlin, J.; Chang, M.W.; Lee, K.; Toutanova, K. BERT: Pre-training of deep bidirectional transformers for language understanding. In Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Minneapolis, MA, USA, 2–7 June 2019; pp. 4171–4186. [CrossRef]
26. Honnibal, M.; Montani, I. spacy 2: Natural language understanding with bloom embeddings. *Convolutional Neural Netw. Increm. Parsing* **2017**, *7*, 411–420.
27. Sharma, V.; Dave, M. SQL and NoSQL databases. *Int. J. Adv. Res. Comput. Sci. Softw. Eng.* **2012**, *2*, 20–27.
28. Bonatti, P.A.; Decker, S.; Polleres, A.; Presutti, V. Knowledge graphs: New directions for knowledge representation on the semantic web (dagstuhl seminar 18371). *Dagstuhl Rep.* **2019**, *8*, 29–111. [CrossRef]
29. Berners-Lee, T.; Hendler, J.; Lassila, O. The semantic web. *Sci. Am.* **2001**, *284*, 34–43. [CrossRef]
30. Horrocks, I.; Patel-Schneider, P.F. Knowledge representation and reasoning on the semantic web: Owl. In *Handbook of Semantic Web Technologies*; Chapter 9; Springer: Berlin/Heidelberg, Germany, 2011; pp. 365–398.
31. Smith, E.M.; Williamson, M.; Shuster, K.; Weston, J.; Boureau, Y.L. Can you put it all together: Evaluating conversational agents' ability to blend skills. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL 2020). Association for Computational Linguistics, Seattle, WA, USA (Virtual), 5–10 July 2020; pp. 2021–2030. [CrossRef]
32. Nenciu, B.; Corlatescu, D.G.; Dascalu, M. RASA Conversational Agent in Romanian for Predefined Microworlds. In Proceedings of the RoCHI-International Conference On Human-Computer Interaction, Sibiu, Romania (Virtual), 22–23 October 2020; pp. 87–94.
33. Vrandečić, D.; Krötzsch, M. Wikidata: A free collaborative knowledgebase. *Commun. ACM* **2014**, *57*, 78–85. [CrossRef]
34. Auer, S.; Bizer, C.; Kobilarov, G.; Lehmann, J.; Cyganiak, R.; Ives, Z. Dbpedia: A nucleus for a web of open data. In *The Semantic Web*; Springer: Berlin/Heidelberg, Germany, 2007; pp. 722–735.
35. Pérez, J.; Arenas, M.; Gutierrez, C. Semantics and complexity of SPARQL. *Acm Trans. Database Syst. (TODS)* **2009**, *34*, 1–45. [CrossRef]

36. Explosion. spaCy Romanian Models. 2022. Available online: https://spacy.io/models/ro (accessed on 20 February 2022).
37. Barbu Mititelu, V.; Ion, R.; Simionescu, R.; Irimia, E.; Perez, C.A. The romanian treebank annotated according to universal dependencies. In Proceedings of the Tenth International Conference on Natural Language Processing (hrtal2016), Dubrovnik, Croatia, 29 September–1 October 2016.
38. Contributors, U.D. Universal Dependencies Corpora. 2014–2021. Available online: http://universaldependencies.org/u/dep/all.html (accessed on 14 July 2021).
39. Masala, M.; Ruseti, S.; Dascalu, M. RoBERT—A Romanian BERT Model. In Proceedings of the 28th International Committee on Computational Linguistics (COLING), Barcelona, Spain, 8–13 December 2020; pp. 6626–6637. [CrossRef]
40. Avram, A.M.; Catrina, D.; Cercel, D.C.; Dascălu, M.; Rebedea, T.; Păiş, V.; Tufiş, D. Distilling the Knowledge of Romanian BERTs Using Multiple Teachers. *arXiv* **2021**, arXiv:2112.12650.
41. Bărbuță, I.; Cicală, A.; Constantinovici, E.; Cotelnic, T.; Dîrul, A. *Gramatica Uzuală a Limbii Române*; Litera: Chisinau, Republic of Moldavia, 2000 .
42. Güting, R.H. GraphDB: Modeling and querying graphs in databases. In Proceedings of the VLDB, Citeseer, Santiago de Chile, Chile, 12–15 September 1994; Volume 94, pp. 12–15.
43. Eclipse Foundation. RDF4J. 2022. Available online: https://rdf4j.org/ (accessed on 1 May 2022).
44. Ontotext. LoadRDF. 2016. Available online: https://www.graphdb.ontotext.com/documentation/7.2/standard/loadrdf-tool.html (accessed on 2 May 2022).