RASA Conversational Agent in Romanian for Predefined Microworlds

Bianca Nenciu, Dragos Georgian Corlatescu, Mihai Dascalu
University Politehnica of Bucharest
313 Splaiul Independenței, Bucharest, Romania
nenciu.bianca@gmail.com, {dragos.corlatescu, mihai.dascalu}@upb.ro

DOI: 10.37789/rochi.2020.1.1.14

ABSTRACT

Technology is becoming omnipresent in our lives due to its accessibility and ease of use. Conversational agents facilitate interactions in natural language and are frequently employed to perform repetitive tasks in a specific context. We introduce a conversational agent for Romanian built on top of the open-source RASA framework, capable to communicate in predefined microworlds. Two scenarios were considered, namely: a smart home assistant which interprets commands to IoT devices, and an interactive info-point for our university focusing on providing guidance to students. Several enhancements were considered, including an NLP pre-processing pipeline from spaCy and a knowledge graph implemented using Grakn for conceptualizing the information accessible to the agent. Our agent can quickly classify intents and extract entities with high accuracy for a given microworld (F1-score of 97% for the first microworld and 93% for the second). A survey on 10 users showed high satisfaction in terms of the usefulness and the succinctness of the provided information.

Author Keywords

Conversational agent; Natural Language Understanding; Romanian language; Microworlds.

ACM Classification Keywords

H.5.2. Information interfaces and presentation (e.g., HCI): User Interfaces.

I.2.7 Natural Language Processing: Discourse, Language parsing and understanding, Text analysis

General Terms

Natural Language; Text analysis.

INTRODUCTION

Internet of Things (IoT) is a concept that has grown in popularity during the last years due to its utility; numerous day to day objects, such as lightbulbs, can be connected to the Internet and can be switched on and off remotely. Another advantage of IoT is that the interaction with physical objects can be performed using text or voice commands.

Conversational agents are systems that mimic characteristics of human interactions and can have unstructured conversations usually meant to provide information or entertainment to users. Conversational agents are frequently employed in many domains and businesses, such as customer support, sales, marketing, and counseling.

There are multiple frameworks that can be used for implementing conversational agents, but they are mostly available for wide-spread languages, such as English, French, or German. However, there is little to no support for less-spread languages, such as the Romanian language.

This article introduces a conversation agent for Romanian language, capable to communicate in specific contexts (i.e., microworlds). The agent understands intents from the user's input and responds accordingly in Romanian. Note that the methods presented here can be extended to other languages, as well. A microworld can be described as a part of the entire world where the agent lives. It knows the rules governing this small world, can interact with individuals by following those rules, but going outside this context will dramatically decrease the quality of the conversation or even making it inconsistent. Our conversational agent focuses on two different microworlds: 1) a home assistant responsible for controlling IoT appliances available in user's residence; 2) a university info-point to provide student orientation (e.g., guidance on the location of classes or of academic staff).

Two important aspects need to be taken into account when building conversational agents: a) *intent classification* – i.e., the agent should be able to understand what users are saying or what are their interests; and b) *entities detection* – i.e., the agent needs to detect key components from the user's sentence and request missing information. For example, if the user asks "What will be the temperature tomorrow?" the system should be able to understand that the question is about the temperature and also to extract the entities "temperature" and "tomorrow" so that it can respond with the missing information, the actual temperature.

This paper continues with a presentation of the commonly used frameworks for building a chatbot. The following section describes the used corpora, alongside the method employed for building our agent. The paper continues with results in terms of performance and a user survey, followed by conclusions and future leads meant to improve the overall capabilities of the system.

RELATED WORK

Snips [5] is a lightweight dynamic processing pipeline implemented in Rust [20] and Python. Snips can be easily integrated with IoT devices that have limited local resources. Nenciu et al. [16] have extended its pipeline to provide support for the Romanian language.

RASA [2] is a mature open-source framework which contains two main components: RASA Natural Language Understanding (NLU) and RASA Core. The first component is responsible for intent classification and entity detection, whereas the second is the dialogue engine which can be used to implement the conversational agent.

The approach of identifying the intent, as well as discovering corresponding entities, can be performed separately or together. The state of the art model for this task is DIET (Dual Intent and Entity Transformer) [3] implemented in RASA. The model tackles the two problems together and can be trained six times faster than other models, while ensuring accurate results for intent classification and entity recognition. As seen in Figure 1, the DIET model can also receive as input pretrained word vectors from BERT [7], ConveRT or GloVe [17].

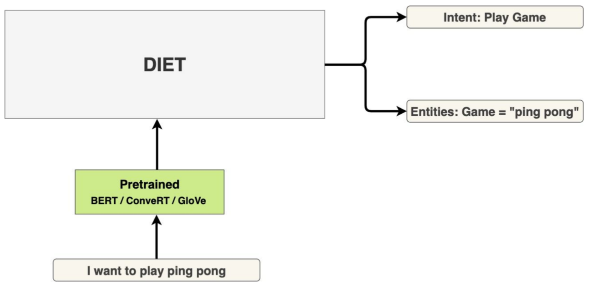


Figure 1. High level illustration of DIET [18].

SpaCy [12] integration is another valuable asset for RASA, especially for low resource languages, such as Romanian. SpaCy is a Natural Language Processing (NLP) tool that provides a common interface for processing all integrated languages. SpaCy can be used to perform part of speech tagging and dependency parsing, whereas these additional insights can help the RASA pipeline provide more accurate results.

In past years, various approaches for simultaneous intent classification and entity recognition have been researched and published. One of the oldest approach was described by Zhang and Wang [26], who used joint models built using Gated Recurrent Unit (GRU) [4], where the hidden state was used for both tasks; their approach managed to outperform the state-of-the-art solutions of that time.

A more recent approach for dual intent classification intent and entity extraction architecture was proposed by Vanzo et al [22] who used both a self-attention mechanism [23] and bidirectional Long Short Term Memory (LSTM) layers [11]. They managed to score better than older versions of Rasa, Dialogflow [8] and LUIS [15].

All previously specified frameworks are focusing on a task-oriented dialogue system. As described by [1] Almansor and Hussain [1], those kinds of systems are solving a specific problem or live in a specific context; this is the reason for being commonly desired by companies. As stated by Sandbank et al. [21], around 80% of interviewed companies want to migrate to this type of solutions in 2020 due to their utility when interacting with a client. The development of such systems is quite straightforward, involving predefined rules and pre-scripted conversations.

For more generic chat bots that are not task-oriented, more advanced solutions are required while relating to their usage scenarios. For example, a chit-chat bot can have issues such as: it can lack specificity, the personality it exposes can be inconsistent, or it can become boring with standard and repetitive responses [25]. State of the art models (e.g., TransferTransfo [24]) for this type of bots consists of approaches using transfer learning and Transformer-based models [23].

METHOD

Corpus

In general, chatbots are task-specific, meaning that they can handle requests from a predefined microworld. This implies that a specific corpus has to be created for each experiment. Two microworlds were explored in this study, namely: a smart home assistant and an interactive info-point for our university. The first corpus was manually created, and it contains 250 sentences on 35 possible intents (see Figure 2 for sample statements). We can further categorize the intents into 12 actions, such as asking about the calendar of the day or controlling home devices. One issue with this corpus was that two intents could have very similar forms, where only one word is different (e.g., "turn the music up" versus "turn the music down"), making it difficult for a model to differentiate between intents.

```
# setTemperature intent
-   Setează temperatura la [roomTemperature](19
    degrees) în [room](bedroom)
-   Poți crește temperatura la [roomTemperature](22
    degrees)?
# getRecipe intent
-   Spune-mi rețeta pentru [recipe](pizza).
-   Găsește-mi rețeta pentru [recipe](clătite).
```

Figure 2. Sample phrases for the first microworld.

The second corpus (see Figure 3) was designed for the university info-point and it consists of both manual and automatically generated sentences. First, we developed a list of entities that can appear in a sentence, such as: name of course subjects (e.g., "Object Oriented Programming", "Electronics"), name of classrooms (e.g., "EG105"), name of teachers, among others. Second, we manually created sentences that had placeholders for the previously mentioned entities. Third, we generated sentences by randomly selecting entities from the specific sets. Given this approach, we generated 80 sentences representing 11 actions which were more different from each other in comparison to the home assistant corpus.

```
## find_schedule_with_course
 -   Unde se desfășoară cursul de [Metode
     Numerice](course)?
 -   Spune-mi, te rog, în ce sală pot participa la
     [Algoritmi Paraleli și Distribuiți](course)
 ## find_schedule_with_class_and_class_type
 -   Unde se ține [laboratorul](class_type) pentru
     grupa [311CB](group_name)?
 -   Unde se ține [cursul](class_type) pentru seria
     [CB](group_name)?
 ## find_schedule_with_course_and_class_and_class_type
 -   Unde se ține [cursul](class_type) de
     [Engleză](course) pentru grupa
     [321CC](group_name)?
```

Figure 3. Sample phrases for the second microworld.

Architecture

The proposed pipeline uses Rasa NLU and corresponding components, and combines them into a new pipeline which offers support for Romanian. We rely on the spaCy model integrated in the ReaderBench framework [6] to perform dependency parsing and part of speech tagging. Figure 4 introduces the overarching pipeline from RASA that relies on spaCy to parse the user query.

The most important components from the NLU engine are the tokenizers – which split the input phrase into smaller semantical units (i.e. words), featurizers – which convert the words into float vectors, and intent classifiers and entity extractors – which in this case are handled all at once by the DIET component. Initial releases of RASA used only a simple CRF (Conditional Random Field) [14], which had problems when the number of training sentences was large (hundreds). DIET has a Transformer-base architecture [23] that uses multiple consecutive CRFs; thus, the new model is no longer susceptible to the initial problems.

Additional relevant Romanian resources integrated in our agent include the DexOnline.ro database, a popular Romanian dictionary. The dictionary itself provides a comprehensive list of word definitions, alongside with word types, popularity, and inflections. Our Romanian resource files consists the following:

- Top 10,000 most used words, together with their corresponding inflections;
- Top 2,000 verbs and lexemes;
- Stop words (i.e., words having no contextual information);
- Randomly generated word lists (i.e., noise used for data augmentation and training the intent classifier);
- Over 1000 of Romanian texts relevant for our microworld scenarios: books, news article, Wikipedia pages.

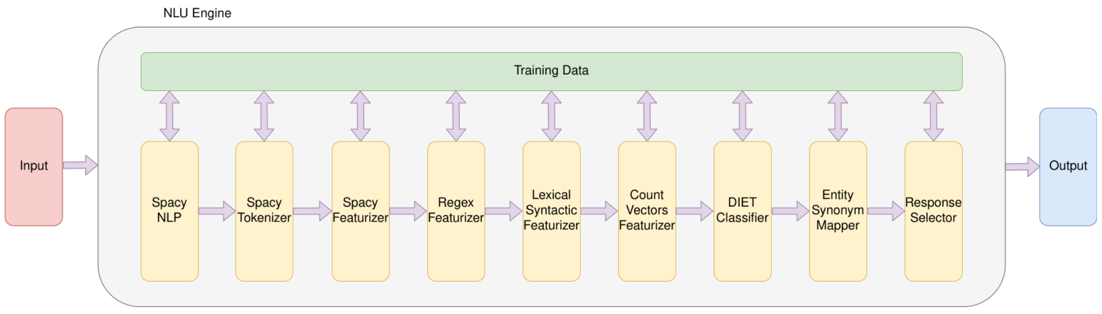


Figure 4. The RASA pipeline integrated with spaCy.

Dialog Management

A dialog manager is responsible for the flow of the dialog between the user and the conversational agent. Figure 5 introduces the steps for the dialog manager, which takes the output of the NLU component, updates the current state of the dialog, the user's history, as well as other important information, and outputs instructions for the response selector.

The input to the dialog manager is a human utterance, converted to its semantic representation by going through the intent classification and the entity extraction process. For example, a question like "Unde găsesc cursul de programare orientată pe obiecte?" (eng. "Where do I find the Object Oriented Programming class?") will be transformed to a query like "find(class='OOP')". As the input is too ambiguous, the dialog management will try to find relevant user information, such as their class name. Furthermore, the knowledge base is queried for information about that specific class and its name. Finally, the agent will output an instruction like "class_location (class_name='2CB', class='OOP', room='PR001')" which is outputted into natural language: "Cursul de programare orientată pe obiecte pentru seria 2CB se ține în sala PR001 la ora 18:00." (eng. "The Object Oriented Programming course for the 2CB series takes place in room PR001 at 18:00.").

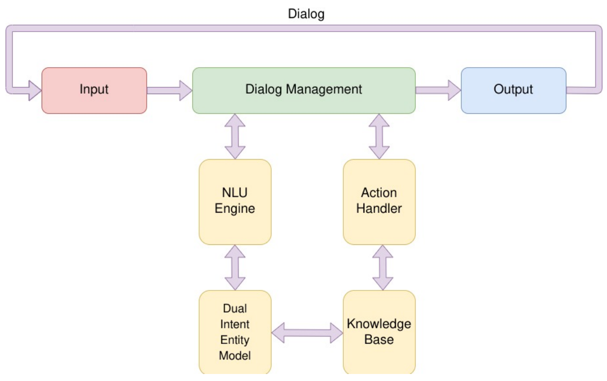


Figure 5. Dialog management architecture.

Intent Classification and Entity Extraction

The first step of every conversation consists of waiting for user's input, which is a characteristic specific to any dialog management system, followed by classifying and extracting entities using DIET. One improvement that can occur at this stage and might be implemented in the future consists of transforming the extracted entities into machine readable representations (e.g., "mâine la 8" / eng. "tomorrow at 8" could be converted to a timestamp).

Context Tracking

User history or previous states must be maintained between queries and replies for conversations to become stateful. For this purpose, a small knowledge base called a "tracker" is built and stored in an in-memory data structure offered by a Redis [19] backend. When a new session starts, a token is randomly generated, which is afterwards passed along with each parsed input. For an even better tracking, the user's username, name, or any form of identification can be passed. In the context of the university chatbot, the tracker can fetch information about courses and other public information after an initial authentication which consists of stating your name.

Response Handling

Response handling is the last, but one of the most important components in building conversational agents. It takes all the information that has been parsed by the NLU engine and previous information held by the tracker, and builds a meaningful answer. There are multiple alternatives in which an agent can produce a reply, which can even involve natural language generation. However, we focused on two simpler techniques: predefined responses and custom actions.

Predefined Responses. The agent can also be trained to associate arrays of predefined responses with intents similar to how it is trained with various input phrases and queries, Moreover, responses do not need to be static, in the sense that the sentences may differ for a set of queries. The simplest strategy for making the conversation more human-like is to define multiple responses for the same intent and randomly select one of those. In addition, placeholders can be automatically replaced based on the extracted entities. For example, if the user greets the agent, then it replies with a greeting as well. A common interaction could be started by the user with a "Hei!" (eng. "Hey!" message, while the agent would respond with "Hei. Cu ce te pot ajuta?" (eng. "Hey! How may I help you?").

Custom Actions. Conversational agents can reply using a custom action implemented in a given programming language that follows an imposed application logic. The usual problem with this approach is that the interactions often seem unnatural, as there is very little nondeterminism or randomness in the output. For this specific reason, custom actions may be combined with predefined responses to reply to the user. Another use case for custom actions is when additional information is needed from the user, or when third party APIs are queried.

Knowledge Representation

Our conversational agent needs to store and retrieve relevant information, as well as the context of a discussion to respond to the user's input. We opted for a non-relational database – Grakn [9] –, an open-source knowledge graph representation that provides an excellent fit for systems operating with highly interconnected data. Grakn provides a concept-level schema which implements the Entity-Relationship model and provides reasoning capabilities. Figure 6 introduces the model corresponding to our second microworld scenario. The agent can perform slot filling tasks by using Graql [10], Grakn's Reasoning and Analytics Query Language, in order to properly continue the conversation with the user.

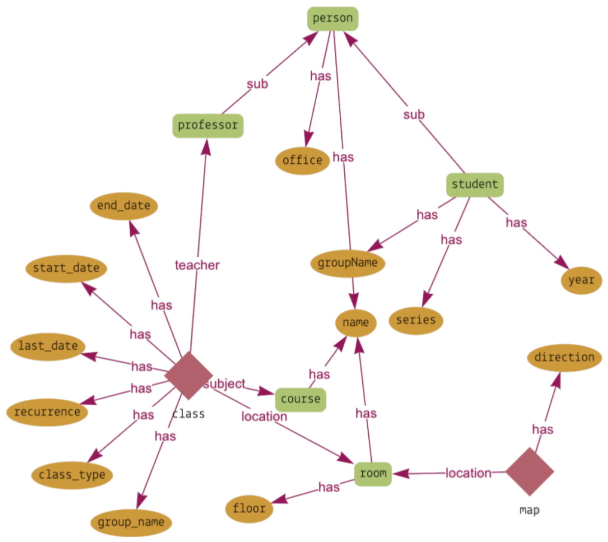


Figure 6. Grakn knowledge graph for the university info-point.

One of the most important and complex parts of the agent while considering the second micro-world relates to navigation queries. The difficulty of answering navigation-related queries comes not only from the absence of a localization system, but also from the fact that a long list of steps may be too difficult to remember. Thus, the agent attempts to describe the destination using the surrounding environment, points of interest, or any other information that the user may already have. Some directions, such as the floor where the room is found, are more important than others, which describe minor details from the surrounding area (see Figure 7).

```
floor sub attribute,
 datatype string;

room sub entity,
 has name,
 has floor,
 plays location;

direction sub attribute,
 datatype string;

map sub relation,
 relates location,
 has direction;

$pr-001 isa  room, has name "PR 001", has floor
"parter";

$map-pr-001 (location: $pr-001) isa map,
 has direction "vis-a-vis de grupurile sanitare",
 has direction "accesibil din holul principal";
```

Figure 7. Schema definition for a point of interest.

Sample conversation

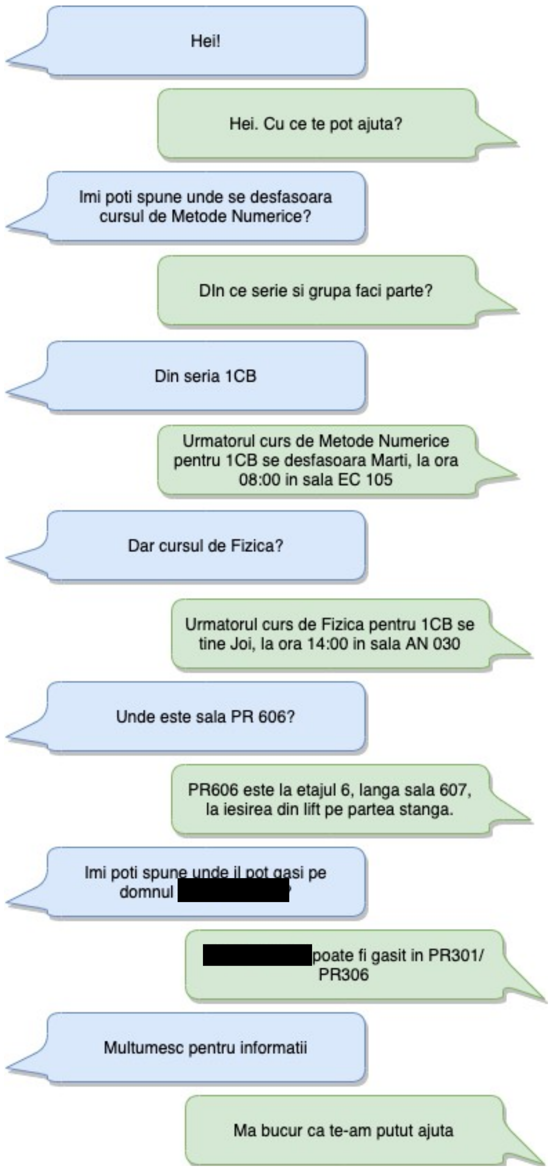
Figure 8 introduces a sample conversation between a user and our conversational agent.

RESULTS

Performance

The training of our agent for each microworld was done on 80% of the corpus; the remaining 20% was used for testing. Although the number of examples is small, this is not a problem due to the characteristics of a microworld which is self-contained, and it considers similar ways to express an intent. The considered metric for assessing the performance of our system was the F1-score.

The first microworld contained 203 phrases belonging to 35 categories, while our test suite contained 94 phrases from the same categories. The system achieved a 97% F1-score with the corresponding confusion matrix from Figure 9. The confusion matrix itself shows promising results and most test phrases were correctly classified. Some pairs of intents which were difficult to classify consist of: "playSong" with "previousSong", or "volumeUp" with "volumeDown. For example, "previousSong" had the lowest F1 scores of 75%, partially because it is an intent with very few phrases which can be mixed up with "playSong". This is somewhat expected, given how similar two inputs are, with the only difference between them being "on" versus "off". In a production environment, we recommend keeping logs for intents and periodically updating the dataset to further increase the system's accuracy and its performance. This can be either done automatically by using clustering algorithms or by manually curating the logs.



Hei!

Hei. Cu ce te pot ajuta?

Imi poti spune unde se desfasoara cursul de Metode Numerice?

DIn ce serie si grupa faci parte?

Din seria 1CB

Urmatorul curs de Metode Numerice pentru 1CB se desfasoara Marti, la ora 08:00 in sala EC 105

Dar cursul de Fizica?

Urmatorul curs de Fizica pentru 1CB se tine Joi, la ora 14:00 in sala AN 030

Unde este sala PR 606?

PR606 este la etajul 6, langa sala 607, la iesirea din lift pe partea stanga.

Imi poti spune unde il pot gasi pe domnul [redacted]

[redacted] poate fi gasit in PR301/ PR306

Multumesc pentru informatii

Ma bucur ca te-am putut ajuta

Figure 8. Sample conversation between a human (left side) and our agent (right side).

The second microworld contains 80 phrases belonging to 11 intents, and our test set contained 27 phrases covering all intents. This microworld included eight predefined dialogs using 23 responses from 9 categories. The predefined dialogs and responses were used to train the response selector. The agent achieved a 93% F1-score and the corresponding confusion matrix is depicted in Figure 10.

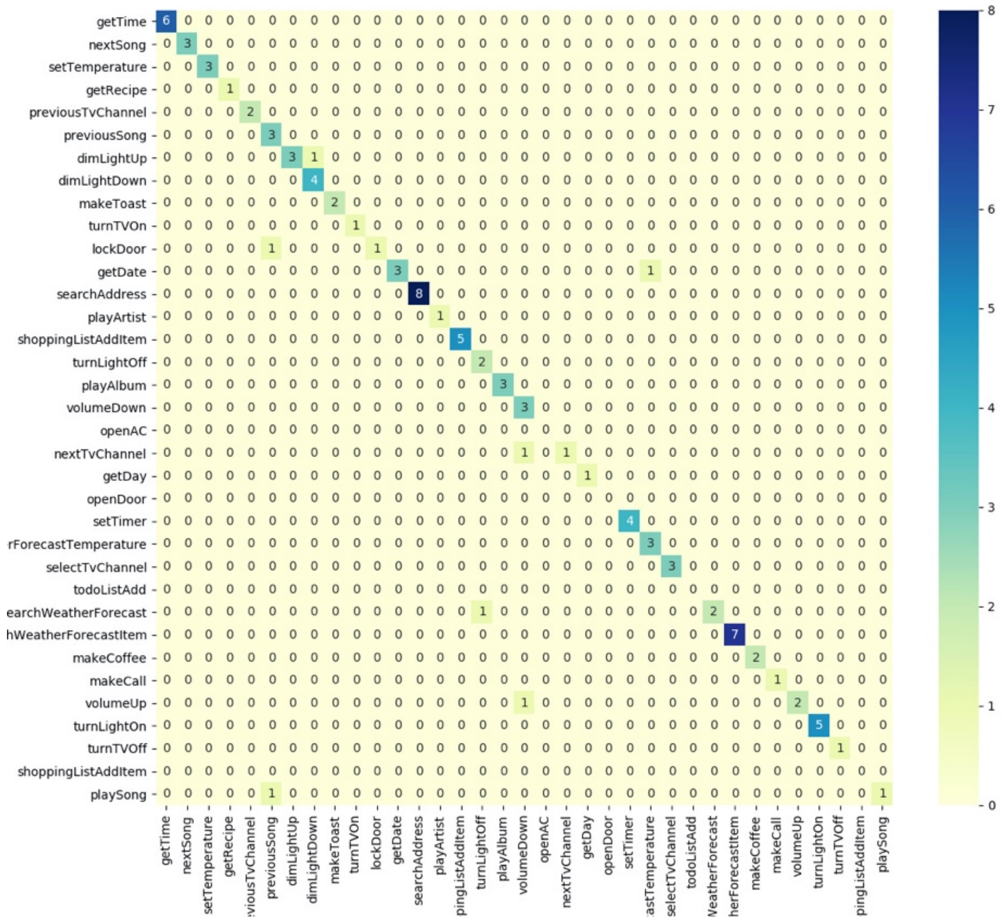


Figure 9. Confusion matrix for the first microworld (home assistant).

Fewer problems are identified since the training dataset was stricter, with fewer categories and with little overlap between the intents. We notice that the "find schedule with course and class and class type" intent is overlapped with "find schedule with class and class type" (without the actual "class"); nevertheless, this does not matter in practice because both intents are handled by the same action. Therefore, the end the user receives the same expected answer.

In addition, the agent has to provide near real-time responses to ensure the flow of the conversation. We achieved response times of less than two milliseconds, which can guarantee the naturalness of the conversation.

User Survey

The NLU engine was evaluated using a small group of 10 undergraduate and Master degree students from our university who queried the conversational agent for information, using the second microworld scenario. The users were afterwards asked to rate their interaction on a Likert scale from 1 to 10 based on the following criteria:

1. How useful was the information? (1 – "Not usefully at all"; 10 – "Extremely useful"; M = 9.00, SD =0.77);
2. How pleasant was the interaction? (1 – "Completely unpleasant"; 10 – "Extremely pleasant interaction"; M = 9.40, SD =0.77);
3. Have you ever considered the other conversation party was a machine? (1 – "I thought I was talking with a person"; 5 – "I cannot say"; 10 representing "I knew that I was talking with a chat bot" M = 5, SD =0.87).

The Intraclass Correlation Coefficient [13] is 0.888, which suggests strong agreement between the replies to the survey. All individuals found the information to be very useful, with a high user satisfaction (9 on a 10-point scale). Most users were satisfied in terms of the quality and the succinctness of the information. The less satisfied users reported they were looking for additional information and they would have preferred to avoid the necessity of a secondary query. While relating to the pleasantness of the conversation, part of the users considered the agent should have included some chit-chat messages, while others considered it to be a very pleasant and the dialog was natural; thus, the low ratings to the last question.

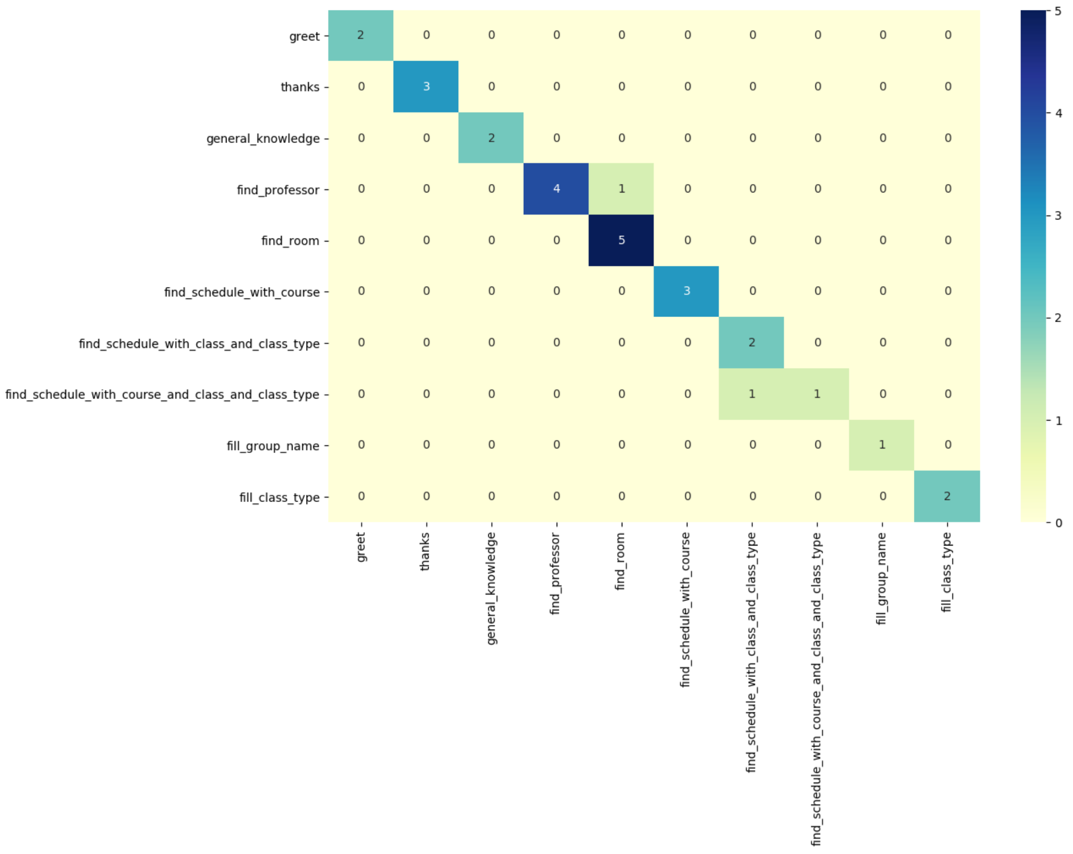


Figure 10. Confusion matrix for the second microworld (university info-point).

CONCLUSION

This paper introduced an NLU engine for Romanian built on top of RASA, capable to quickly classify intents and extract entities with high accuracy for a given microworld. The results are promising for small microworlds that contain a limited number of phrases used to express an intent, and most alternatives are similar. We are working towards building and testing on a larger corpus, which should result in a more general system.

In contrast to close source alternatives, our project runs locally, and it requires few resources after the DIET classifier was trained. Intent classifying with an external service could be easier to implement, but the processing would take a considerably longer time because even the ideal round trip time could already be over 20 times slower than the usual processing time of our engine (1-2ms).

We consider that an important improvement for the NLU engine consists of integrating advanced language models (e.g., a Romanian BERT model), which would expand further the agent's capability across microworlds. In addition, we plan to expand our research to corpus-based architectures to ensure more natural conversations.

Another interesting area of research relates to the classification of multiple intents from a single user statement. In real world situations, we often find ourselves building complex phrases containing multiple actions. The current system returns only the most probable action or none, if no probability is greater than the imposed threshold. In tight correlation to the previous research lead of using the agent in real-life scenarios, our approach was evaluated independently from a speech-to-text engine, which would induce additional errors; however, our training set only contains correct phrases, with no spelling errors. Thus, additional fine-tuning, integration of correction mechanisms, and extensive testing are required.

ACKNOWLEDGMENTS

This work was supported by a grant of the Romanian National Authority for Scientific Research and Innovation, CNCS – UEFISCDI, project number PN-III 72PCCDI ⁄ 2018, ROBIN – "Roboții și Societatea: Sisteme Cognitive pentru Roboți Personali și Vehicule Autonome".

REFERENCES

1. Almansor, E.H. and Hussain, F.K., 2019. Survey on Intelligent Chatbots: State-of-the-Art and Future Research Directions. In Proceedings of the Conference on Complex, Intelligent, and Software Intensive Systems (Sydney, Australia), Springer, 534-–543.
2. Bocklisch, T., Faulkner, J., Pawlowski, N., and Nichol, A., 2017. Rasa: Open source language understanding and dialogue management. *arXiv preprint arXiv:1712.05181*.
3. Bunk, T., Varshneya, D., Vlasov, V., and Nichol, A., 2020. DIET: Lightweight Language Understanding for Dialogue Systems. *arXiv preprint arXiv:2004.09936*.
4. Cho, K., Van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., and Bengio, Y., 2014. Learning phrase representations using RNN encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*.
5. Coucke, A., Saade, A., Ball, A., Bluche, T., Caulier, A., Leroy, D., Doumouro, C., Gisselbrecht, T., Caltagirone, F., and Lavril, T., 2018. Snips voice platform: an embedded spoken language understanding system for private-by-design voice interfaces. *arXiv preprint arXiv:1805.10190*.
6. Dascalu, M., Dessus, P., Trausan-Matu, Ş., Bianco, M., and Nardy, A., 2013. ReaderBench, an environment for analyzing text complexity and reading strategies. In Proceedings of the International Conference on Artificial Intelligence in Education (Memphis, TN, United States), Springer, 379–388.
7. Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K., 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.
8. Google, 2020. Dialogflow. Retrieved September 30th 2020 from https://cloud.google.com/dialogflow.
9. Grakn, n.d. Grakn webpage. Retrieved July 27th 2020 from https://grakn.ai/.
10. Graql, n.d. Graql Query Language. Retrieved July 27th 2020 from https://dev.grakn.ai/docs/query/overview.
11. Hochreiter, S. and Schmidhuber, J., 1997. Long short-term memory. *Neural computation 9*, 8, 1735–1780.
12. Honnibal, M. and Montani, I., 2017. spacy 2: Natural language understanding with bloom embeddings. *convolutional neural networks and incremental parsing 7*, 1.
13. Koch, G.G., 1982. Intraclass correlation coefficient. In *Encyclopedia of Statistical Sciences*, S. Kotz and N.L. Johnson Eds. John Wiley & Sons, New York, NY, 213–217.
14. Lafferty, J., McCallum, A., and Pereira, F.C., 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In Proceedings of the ICML (Williamstown, MA, USA), Morgan Kaufmann, 282–289.
15. Microsoft, 2020. LUIS. Retrieved September 30th 2020 from https://www.luis.ai/.
16. Nenciu, B., Ruseti, S., and Dascalu, M., 2018. Extracting Actions from Romanian Instructions for IoT Devices. In Proceedings of the 13th Int. Conf. on Linguistic Resources and Tools for Processing Romanian Language (ConsILR 2018) (Iasi, Romania), 168–176.
17. Pennington, J., Socher, R., and Manning, C.D., 2014. Glove: Global vectors for word representation. In Proceedings of the Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP) (Doha, Qatar), Association for Computational Linguistics, 1532–1543.
18. RASA, 2020. Introducing DIET: state-of-the-art architecture that outperforms fine-tuning BERT and is 6X faster to train. Retrieved July 27th 2020 from https://blog.rasa.com/introducing-dual-intent-and-entity-transformer-diet-state-of-the-art-performance-on-a-lightweight-architecture/.
19. Redis, n.d. Redis Homepage. Retrieved July 27th 2020 from https://redis.io.
20. Rust, n.d. Rust documentation. Retrieved July 27th 2020 from https://prev.rust-lang.org/en-US/.
21. Sandbank, T., Shmueli-Scheuer, M., Herzig, J., Konopnicki, D., Richards, J., and Piorkowski, D., 2018. Detecting egregious conversations between customers and virtual agents. In Proceedings the 2018 Conference of the North American Chapter of the Association for Computational Linguistics, Volume 1 (New Orleans, Louisiana), ACL, 1802–1811.
22. Vanzo, A., Bastianelli, E., and Lemon, O., 2019. Hierarchical multi-task natural language understanding for cross-domain conversational ai: HERMIT NLU. *arXiv preprint arXiv:1910.00912*.
23. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., and Polosukhin, I., 2017. Attention is all you need. In Proceedings of the Advances in neural information processing systems (Long Beach, CA, USA), Curran Associates, Inc., 5998–6008.
24. Wolf, T., Sanh, V., Chaumond, J., and Delangue, C., 2019. Transfertransfo: A transfer learning approach for neural network based conversational agents. *arXiv preprint arXiv:1901.08149*.
25. Zhang, S., Dinan, E., Urbanek, J., Szlam, A., Kiela, D., and Weston, J., 2018. Personalizing dialogue agents: I have a dog, do you have pets too? *arXiv preprint arXiv:1801.07243*.
26. Zhang, X. and Wang, H., 2016. A joint model of intent determination and slot filling for spoken language understanding. In Proceedings of the IJCAI (New York, New York, USA), AAAI Press / International Joint Conferences on Artificial Intelligence, 2993–2999.