Learning from Limited Datasets: Implications for Natural Language Generation and Human-Robot Interaction

Jekaterina Belakova*
Kungshamra 71
Solna, 170 70
Sweden
ekaterinabeljakova@gmail.com

Dimitra Gkatzia
10 Colinton Road
Edinburgh, EH10 5DT
Edinburgh Napier University
d.gkatzia@napier.ac.uk

Abstract

One of the most natural ways for human robot communication is through spoken language. Training human-robot interaction systems require access to large datasets which are expensive to obtain and labour intensive. In this paper, we describe an approach for learning from minimal data, using as a toy example language understanding in spoken dialogue systems. Understanding of spoken language is crucial because it has implications for natural language generation, i.e. correctly understanding a user's utterance will lead to choosing the right response/action. Finally, we discuss implications for Natural Language Generation in Human-Robot Interaction.

1 Introduction

Robots are becoming prevalent as the technology advances and the prices drop. The International Federation of Robotics[1] reported that in 2017, there was a worldwide increase of 30% for industrial robots sales and there is a 39% increase of professional service robots the sales (in value), while forecasting a growth of 30-35% per year until 2020 for domestic robotics. This will create opportunities for effective human robot communication and will require robots to combine different skills such as computer vision, language understanding and generation as well as object manipulation.

Human-robot interaction (HRI) can be enhanced via the use of natural language dialogue between humans and robots. In this paper, we discuss the implications of dialogue for HRI, by deriving insights from recent work on personal assistants. In particular, we describe how `one-shot learning` can guide natural language generation in scenarios where we only have access to small amounts of example dialogues and discuss how we can transfer lessons learnt to human robot communication. Therefore, we initially describe the development of a personal assistant capable to handle users' queries without being trained with example dialogues, and then we describe how we can adapt this approach to human-robot communication.

2 Approach

MOOBO is a personal assistant for an educational platform Moodle[2] that takes as input users queries (such as queries regarding coursework, dealines, etc.) and outputs responses. Moodle is used by a large number of universities and it allows lectures to share their learning materials such as slides, academic papers, laboratory work as well as coursework and assignments. The students can then access all these documents and posts for their courses. This data becomes available in both a structured and unstructured way. MOOBO is able to access this data and extract the relevant information and render it to users in natural language.

2.1 Software Architecture

MOOBO is a web-based, platform independent application and available to use on all devices: desktops, tablets and mobiles. It uses a client-server architectural style which consists of two components, the client and the server, as shown in Figure 1. The client makes a call to the server and gets the response back. The server is contin-

*This work was completed while Jekaterina was a student at Edinburgh Napier University.

[1] https://ifr.org/

[2] https://moodle.org/

Proceedings of the Workshop on NLG for Human–Robot Interaction, pages 8–11,
Tilburg, The Netherlands, November 8, 2018.

uously listening to client requests. They communicate over HTTP using REST methods (such as GET, POST, PUT, DELETE) in a JSON format.

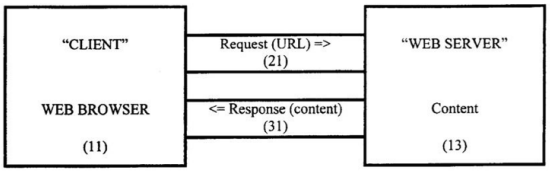

Figure 1: Client-server architecture

The client is a web browser passing on the user input to the server. It is developed using JavaScript framework, HTML and CSS. The server is developed in Python using Flask web framework that offers a development server and RESTful request dispatching.

MOOBO is effectively a spoken dialogue system and thus, it consists of five main components which are responsible for: Speech Recognition, Natural Language Understanding, Dialog Manager, Natural Language Generation and Text-to-Speech Synthesis as shown in Figure 2.

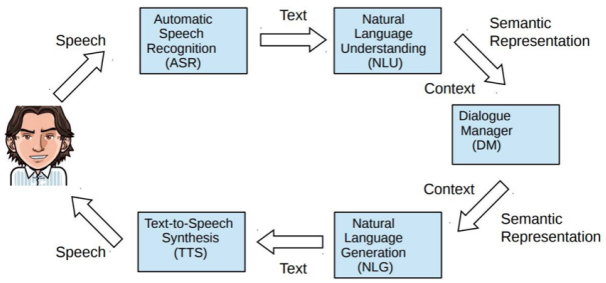

Figure 2: MOOBO's architecture.

Speech Recognition Speech Recognition uses a JavaScript library called `artyom.js` [3]. It resides on the client side and listens to the users input which is then forwarded to the server for further processing. To improve the user experience, both speech recognition and an option to write the input as a text are available.

Natural Language Understanding To process the user input, `spaCy` [4] was used in order to recognise Named Entities and part of speech.

Dialogue manager The Dialogue Manager (DM) is responsible for choosing the action which will lead to generating output. For this domain, dialogues were not available and therefore we created a small dataset of potential dialogues. Then each utterance was mapped to an intent as seen in Table 2. The main challenge that the dialogue manager needed to address is that different students ask for the same information in different ways. For instance, a student can ask "What is the module about?" and "What will I learn from the module?". Although these questions are phrased differently, the intent is the same: the student is requesting a module summary. When several examples of dialogues are available, it is easy to learn that both questions result in the same intent. However, when we only have one example of an intent, we need a clever way to associate all similar queries to this one example. Therefore, we used one-shot learning (Schroff et al., 2015) to address this challenge.

One-shot learning One-shot learning initially learns an embedding per instance usually using some deep learning approach. Once the embeddings have been produced, then the intend recognition simply becomes a k-NN classification problem. In our setup, one-shot learning was achieved as follows:

1. Utilising the knowledge of NER and part of speech tagging, embeddings of the natural language utterances were created using `Word2Vec` (Mikolov et al., 2013) with a 4-word window.

2. The K Nearest Neighbour algorithm (K-NN) was used to find the nearest utterance in the small dataset in terms of the Euclidean distance. After the Euclidean Distance is calculated, the system selects the three closest results and sorts them in terms of distance and selects the first one.

Because K-NN can be sensitive to outliers and has no confidence, the application used three nearest neighbours to make the result more stable.

There are six tasks that the system can perform as depicted in Table 2. They all require either information extraction or text summarization. This is different to traditional dialogue systems which utilise structured information stored in databases.

2.1.1 Natural Language Generation

After the DM has identified the right task, it sends it to the Natural Language Generation (NLG)

[3] https://sdkcarlos.github.io/sites/artyom.html
[4] https://spacy.io/

Input	Intent
What can I potentially learn from the module	module_summary
What is the coursework summary	cw_summary
What are my courses	course_summary
Who is the programme leader for the module	programme_leader
When is the coursework deadline	cw_deadline

Table 1: Examples of utterances mapped to intents.

Task Management
1. Coursework summary
2. Coursework deadline
3. Module summary
4. Course summary
5. Get a program leader
6. Lab/ Lecture summary

Table 2: List of MOOBO's actions.

module. At this instance, NLG is template-based with slot-filling.

Slot-filling in our project, required accessing unstructured text and deriving the correct information. Consider for instance the task of finding a program leader. The Named Entity Recognition module is used to look for a PERSON entity in a specific module section. The coursework deadline was extracted using Spacy NER DATE and ORGINAL types. Some coursework files were written in the specific template, which gave a possibility to use regular expressions to extract the information. For summaries generation TextRank was used (Mihalcea and Tarau, 2004). TextRank is a graph-based ranking algorithm which builds a graph, where the vertices are the units (extracted sentences) to be ranked. The algorithm measures the similarity between the sentences and attaches a ranking score to each one of them.

Figure 3 shows MOOBO's interface and a short example of dialogue.

3 Evaluation

The system was evaluated with humans through a task-based evaluation, followed by a questionnaire. There were 18 participants recruited who are all undergraduate students at Edinburgh Napier University (so they were all familiar with the standard Moodle). Each participant was given a general overview of MOOBO and time to interact with the system. Each user was tasked to perform

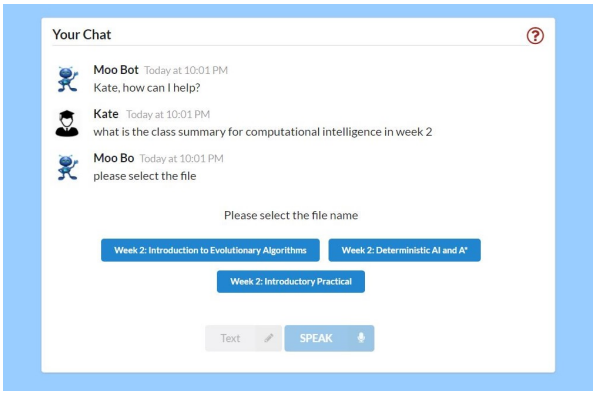

Figure 3: MOOBO's interface.

Questions
1. Was MOOBO accurate?
2. Was MOOBO easy to use?
3. Would you use MOOBO
4. Would you prefer using Moobo or a standard Moodle?
5. Overall how would you rate the experience? (0-bad, 10-excellent)

Table 3: List of questions answered by participants after completing the task-based evaluation.

a set of pre-defined tasks using MOOBO and then using the standard Moodle. Specifically, the participants had to find information regarding the following:

1. The lab summary for "fundamentals of parallel systems" in week 2.
2. The coursework for "computational intelligence".
3. The deadline for the "Algorithms and Data structures" module.
4. The program leader for the "Design Dialogues" module.

After finishing these tasks, the participants were given a short questions (see Table 3.

4 Results and Discussion

The results showed that the participants really preferred MOOBO to standard Moodle. In fact, 76% of students said that it was accurate, 24% mentioned that it was accurate to an extent, adding that "I had to repeat a few times, but it was accurate afterwards" and "Sometimes it was unable to recognize what I said". Interestingly, none of the participants said that MOOBO was inaccurate.

All participants said that MOOBO was easy to use, which was expected given the widespread use of personal assistants nowadays as well as the participants' background. 71% of the users said they would use MOOBO, with 47% answering that they would use Moobo over Moodle. 24% stated they would use both, depending on the task and only 29% preferred the standard Moodle.

In the last question, students were asked to rate the overall experience from 0 to 10, where 0 is bad and 10 is excellent. The average rating was 8.5 ($mode = 8$, $median = 8$, no rating below 7 was given).

As seen from the results, Personal Assistants are positively seen by the users and they can speed up and ease performing specific tasks. Most students (76%) said that the answers were accurate which shows that the question was understood, and the Dialogue Manager selected the correct intent. However, there were some misunderstandings and MOOBO could not recognise the words or allocate the right task for the input. The second question received overwhelming responses. Every tester said it was easy to use MOOBO. This means that the designed user interface helped with the interaction. Extra features such as providing the link to a requested file and re-confirming if the question is correct were highly valued by users and helped them to access the information quicker. Personal Assistants become more popular and used, however they are not completely integrated with daily tasks.

5 Discussion and Conclusions

From the results presented, the following conclusions can be drawn for real-world NLG systems. Firstly, NLG NLG for interactive systems is an extremely challenging task. The main reason for this is that NLG is always influenced by other factors, such as natural language understanding, object recognition, human action recognition, dialogue management etc.

Secondly, NLG is quite domain-dependent, which requires access to example datasets of dialogues and interactions or access to experts. Both can be very expensive to acquire. By using approaches such as `one-shot learning` or even `zero-shot learning`(e.g. (Sadamitsu et al., 2017)) can help reducing the need of acquiring sizeable datasets. Our proposed setup can be extended to include visual information, which will enhance a robot's capability to monitor the environment and allow it to refer to objects in it as well as reason about it.

Finally, our toy example shows that we can approximate the state of the system by using embeddings. Pre-trained embeddings transfer knowledge from other domains to a new one and are especially useful in situations where only small datasets are available. This is an approach that can be transferred to human-robot communication. For instance, in situated setups, where a human and robot work together to accomplish a task such as assembling furniture, image and language embeddings can be used to approximate states, even if these states do not exist in the dataset.

6 Summary and Future Work

Acknowledgments

We are grateful to Edinburgh Napier University's technical team for granting us access to the university's Moodle environment as well as all the modules leaders who kindly shared and allowed us to use all their learning resources.

References

Rada Mihalcea and Paul Tarau. 2004. Textrank: Bringing order into text. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Neural and Information Processing System (NIPS)*.

Kugatsu Sadamitsu, Yukinori Homma, Ryuichiro Higashinaka, and Yoshihiro Matsuo. 2017. Zero-shot learning for natural language understanding using domain-independent sequential structure and question types. In *Proc. Interspeech 2017*, pages 3306–3310.

Florian Schroff, Dmitry Kalenichenko, and James Philbin. 2015. FaceNet: A Unified Embedding for Face Recognition and Clustering. pages 815–823.