Automated Questions for Chat Dialogues with a Student Office Virtual Agent

Mikulas Muron[1,3] and Kristiina Jokinen[1,2]

[1]University of Tartu
[2]University of Helsinki
[3]Mendel University in Brno

Abstract. The paper describes work on building a virtual agent dialogue system for the purpose of providing information that is relevant for new university students. Although the system is meant to be used for Student Office dialogues, it can easily be also adapted to other similar information providing systems due to being guided by a focus on reusability in design and easy adaptation to other users and situations. The work uses the Virtual Human Toolkit, and the main motivation has been to provide a simple and useful way to automatically transform information from websites into a dialogue with a virtual human.

Keywords: Virtual Agent, Chat Dialogues, Web-based WikiTalk

1 Introduction

Dialogue technology is already widely used to provide useful information concerning various tasks and enable chat-like interactions related to topics that are of interest to the user. Applications such as Siri, Cortana, Now, and Alexa are used as examples that successfully combine advanced speech and language technology to mobile applications, and apparently almost 20% of all Google searches are accomplished via voice. This makes conversational interfaces of big need, both from the research and development point of view, concerning easy-to-use, natural, flexible, affective, and adaptable user interfaces (see e.g. McTear et al. 2016).

Another line of research in spoken conversations and dialogue management concerns robot and animated agent interfaces, where the human-like appearance calls for more human-like multimodal communication, and the research and development has focused on social interaction, open-domain conversations, multimodal issues, and affective computing for companion applications. Open-domain interactions can be chatbot type conversations where the user's interest drives the system's presentation from one topic to another. For instance, the WikiTalk application (Wilcock and Jokinen, 2013, Jokinen and Wilcock 2014) is a multilingual spoken dialogue system that runs on a Nao robot, and the user can have a dialogue with the robot in which the robot talks fluently about an unlimited range of topics using information from Wikipedia.

Chat-bots and virtual agents have made conversational AI-agents common and useful devices for various tasks where interaction with the user is needed. They can be used as chat-based interfaces to give directions, answer questions, guide virtual tours, etc., as well as to provide training, companionship, and entertainment. The present work studies virtual agents and interaction involving information giving dialogues, and it discusses various steps in building a dialogue system, especially the possible user questions that the system can understand, on the basis of the information found on the internet web pages. The application area is a Virtual Student Office Agent which can provide orientation information for new university students.

The current work extends the WikiTalk open-domain dialogue management towards creating dialogues from any webpage. On the other hand, it also restricts the WikiTalk system by focusing on a particular domain and task-oriented dialogue: the goal is to help new students in their orientation to studies by providing useful information based on the relevant topics already designed on the webpage. Unlike WikiTalk, which deploys the user's interest in the current discussion topic as the driving force for the conversation, the Student Office Agent functions in a more traditional manner, and tries to anticipate possible user questions so as to be able to offer truthful information and reliable answers to the user.

The virtual agent is built on the basis of Virtual Human Toolkit (Hartholt et al. 2013), and it uses the University of Tartu webpages. The main guiding principle is that the system should easily transform information from any website into a dialogue model with a virtual human, and thus be automatically adapted to other information providing domains. For this purpose, the focus is on reusability in design and easy adaptation to users and situations.

The paper is structured as follows. We first describe background for the work and present the problem in Section 2, then continue with the system overview in Section 3, with detailed discussion of the methods and techniques used in the system. Examples and screenshots of the interaction are given in Section 4, and finally we conclude in Section 5.

2 Background and problem presentation

When building dialogue systems, one of the important issues is to equip the system with appropriate and sufficient domain information. This determines the type of questions the user can ask and the details of information that the system can talk about. The domain information needed for dialogue systems often exists in the form of a website, so the question is how to use this information for creating dialogues with humans. In this paper, first steps are described to automatically transform information from relevant websites into natural language dialogues. Related work has been described e.g. by Feng et al. (2006), but our work differs from this in that we have mostly used open source code in our implementation. Integration of open source tools can be highly beneficial for the research community. Moreover, our goal is to enable conversational interactions with virtual agents in the framework of Virtual Human Toolkit (Hartholt et

al. 2013), and for this, we have focused especially on user questions and their generation based on the information in pertinent websites. The Virtual Human is to give an answer to the user by linking the question to an appropriate website which contains the relevant information.

The Virtual Human (VH) Toolkit (https://vhtoolkit.ict.usc.edu/) is a software for creating, researching, and testing of 3D virtual agents with both text and speech capability. Dialogue management is handled by the NPCEditor (Leuski and Traum 2011), which is used for designing dialogues, and for linking environments and non-verbal action. It uses utterance-response pairs to calculate probabilities that represent how likely each response is given a particular input. When choosing a response to the user's behavior, it combines knowledge about the word distributions in user utterances and information about the word distributions in possible responses. Such sophisticated statistical models can lead to good performance, but their construction also requires a large number of training examples regarding representative query-response pairs as well as examples that content-wise cover the whole query space, i.e. all words and expressions that may occur in conversation with the Virtual Human. In order to build a robust dialogue manager it is thus necessary to obtain a fairly comprehensive list of possible questions that the user may ask when conversing with the Virtual Human, so as to pin down the topics that the interactions can be about.

For instance, a Virtual Human aiming to help new university students to orientate with their studies will encounter a myriad of questions related to information on courses, curricula, grants, etc., and should thus be prepared for conversations that deal with such information. In particular, in order to enable students to conduct information seeking dialogues in an easy and natural manner, the Virtual Human should provide friendly and reliable interaction where possible user questions are related to the websites that contain the relevant information that the students are looking for.

The challenge addressed in this paper is how to automatically generate suitable user questions for a Virtual Human chat-system, built within the VH Toolkit framework. Although suitable questions and dialogues can be designed manually, automatic generation would allow a quick and general method to create information seeking dialogues for any domain with the relevant information in a website. The challenge can be divided into three subtasks:

1. how to determine the meaningful part of a given website,
2. how to extract important keywords, and
3. how to generate the questions based on the keywords.

These tasks will be discussed in more detail below.

3 System overview

We must first emphasize that the algorithms used to complete each of the three subtasks are well-known algorithms with mostly open source code. However, the unique contribution of the paper is the integration of these existing algorithms into a single working platform which does not only provide a better fitting solution to our particular Student

Office Virtual Agent application, but also raises several interesting questions concerning system integration as well as system emergence and complexity.

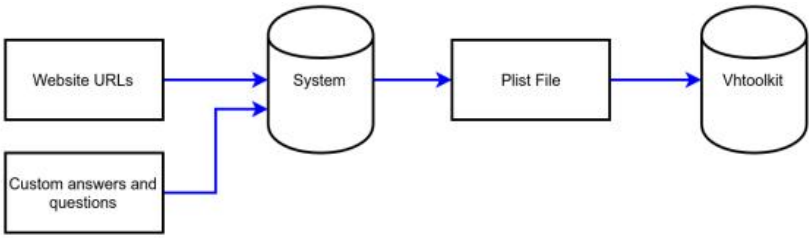

Fig. 1. A black-box view of the system.

A black-box view of the system architecture is presented in Figure 1. The input to the system is a list of URLs from which the dialogue is to be created. The user may also define own custom pair of answers and questions which will be added as extra dialogue data. The data is then processed by the system, and the Property list (`Plist`) file, containing the dialogue itself, is generated as the output. This is further integrated with the VH Toolkit to enable conversations with a Virtual Human.

The system itself contains several components which operate sequentially, and some of the components share the same data. An overview of the system components is given in Figure 2.

3.1 Meaningful part of a given website

The first subtask is to select the relevant information from the designated websites. Much research has already published on selecting meaningful parts of a given website, see e.g. Rahman (2001) and Suhit (2013). Also there exists tools and libraries which can be used for the task.

The user specified websites are first downloaded and saved, and the HTML code is loaded to the class `PageDataHolder`. This contains the data from which useful paragraphs and sentences are extracted.

The key ability of the text and paragraph extraction component is to recognize which sections are valid and useful for further processing. As only the relevant part of a given webpage is selected and used in the follow-up steps, it is important to select the relevant parts correctly. For example the navigation menu, footer and other similar entities are discarded as unnecessary. The selection task uses the python library Readability (Yang Hao: https://github.com/kingwkb/readability). The algorithm goes through the stored HTML code and looks for compact amounts of text and the related HTML tags. Each found section is evaluated with a score, and those with a score higher than a threshold are judged as important. An example of a website and the selected important section is shown in Figure 3.

The system then continues to extract paragraphs from the selected section. HTML standard provides tags that mark the start and end of the paragraphs, allowing them to be easily separated. Paragraphs stored in a class `paragraphDataHolder` inside

`PageDataHolder`. In other words, `PageDataHolder` takes care of each page and contains a list of paragraphs stored in `paragraphDataHolder`.

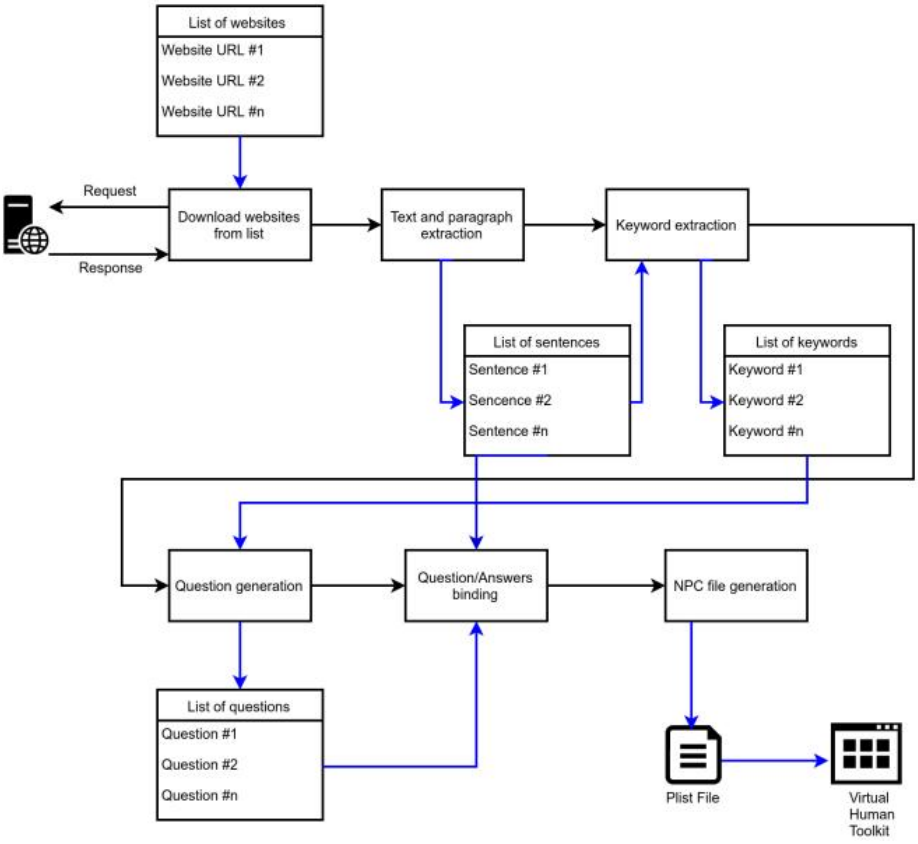

Fig. 2. System component overview. Black arrows indicate control flow, blue arrows data flow.

The last step in the text selection component (before keywords extraction) is to clean up the text. This means that all digits and punctuation characters such as exclamation marks, dots, commas etc. are removed from the text. Also stop words such as *the, and, a*, etc. are removed. This implementation uses the stop words list Natural Language Toolkit provided by the python library (http://www.nltk.org/).

3.2 Keyword extraction

The second subtask is to determine suitable keywords for each paragraph. The keywords represent an overview of the paragraphs and provide a simple estimation of their content. Question formation will be based on the selected keywords, and questions can thus be linked to the relevant paragraph which functions as an appropriate answer.

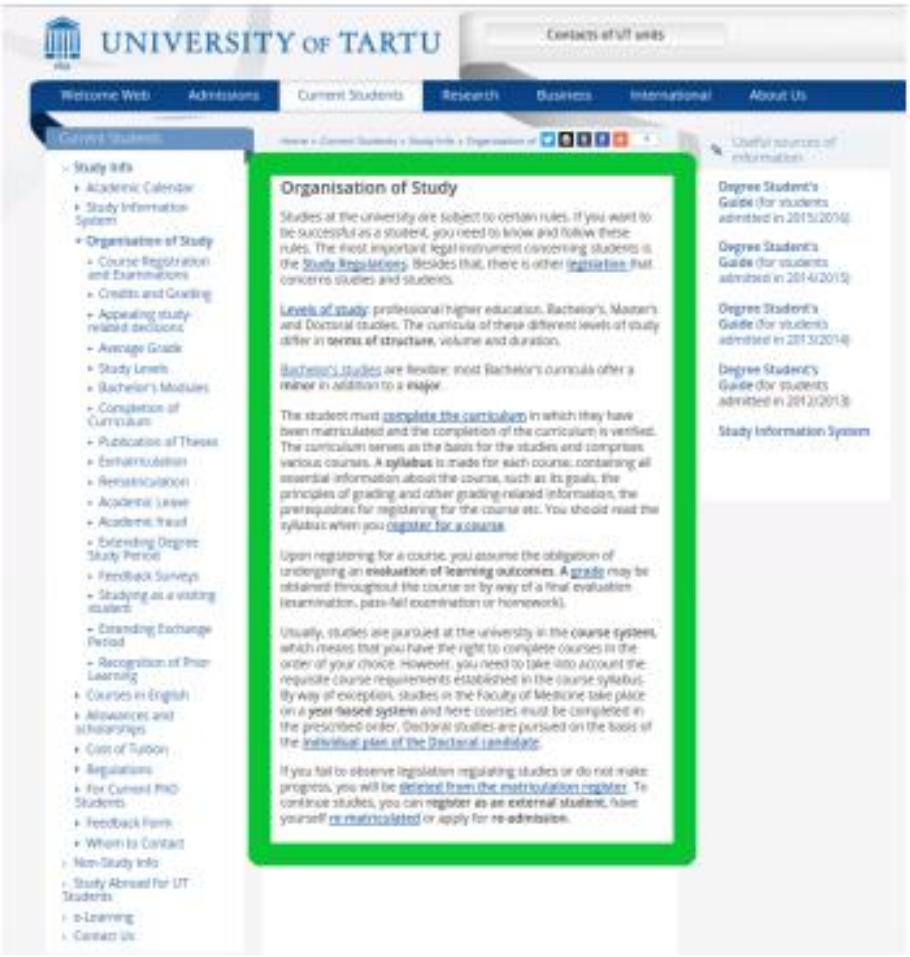

Fig. 3. An example Tartu University webpage. Source: http://www.ut.ee/en/organisation-study

Inspiration for the keyword extraction is drawn from Mooney (2003) and Matsuoka (2003). Two approaches are used: Naive Bayesian filter and TF-IDF metric. Since the performance of the algorithms may vary depending on the paragraph, the designer will try both approaches and chose the best one based on the evaluation of the test cases. Currently, selection of which algorithm to use is made in advance, and the selected algorithm is then run to preform keyword extraction. In this way the designer can select successful keywords and optimize the links between the keywords and possible questions (to be formulated in the next phase), but naturally such manual pre-check are feasible only for small domains. Evaluation of the extracted keywords is needed as a general solution.

Naive Bayesian filter.
The Naive Bayes (NB) algorithm belongs to a family of probabilistic classifiers based on applying Bayes' theorem with strong independence assumptions between the features. It needs a large amount of text for the comparison of relative word frequencies, and for the purposes of this project, books from the Project Gutenberg (such as Jane Austen's Pride and Prejudice and some others) are used for computing the general data. Since this data focuses on novels and fictional texts, it is obvious that higher precision would be achieved if the frequencies were based on domains representing or being close to, what can be found on modern educational websites, or on balanced synchronic statistics, e.g. British National Corpus (BNC XML Edition). Another alternative to generate statistics/n-grams is the approach of Google Books Ngram Corpus (Lin et al. 2012) containing all the books digitized at Google. These alternatives will be explored and deployed in the future.

In this work, a straightforward ratio of word frequencies is used as an importance sorting score, following Jurafsky (2016). The relative frequencies of every word in the sample data (f_s) as well as in each extracted paragraph (f_p) are computed, and the latter is divided by the former. This ratio f_p/f_s measures how frequently a word occurs in the paragraph relative to the normal sample. Words are then sorted according to their importance score, and the best *n* words are used as keywords. In our experiments, *n* is 4 – this is determined on empirical observation by experimenting with different values and this is seems to produce the best results.

For example, given the paragraph in Table 1 (from the University of Tartu website), the obtained keyword scores are those given in Table 2. As can be seen, the keywords seem to describe the core meaning of the given paragraph fairly well.

Table 1. A paragraph from the University of Tartu website (http://www.ut.ee/en/courses-taught-english)

If you arrive after the registration deadline, please contact the dean's office of the faculty to which you are registered to for help with registration. Thus, later arrival does not automatically mean that there is no possibility to join courses, if there are still vacant places available. Do not hesitate to contact us if you need more information about the courses.

Table 2. Keywords extracted from the example paragraph in Table 1.

Keyword	Score
registration	964 488
deadline	482 244
courses	160 748
later	80 374

TF-IDF (Term Frequency – Inverse Document Frequency).
Keywords extraction can also be done by a numerical statistical method which is commonly used in text summarization and ranking the document relevance in search engines (Ramos, 2003). The algorithm consists of two parts: term frequency calculation

(TF - similar to NB algorithm described above) and Inverse document frequency (IDF). The latter is calculated as the logarithm of the number of documents containing a given word divided by the total number of documents. The final TF-IDF metric is then obtained by multiplying TF with IDF.

3.3 Generating Questions

The final subtask for the system is the generation of suitable questions. This is performed on the basis of the extracted keywords. The user can specify a list of template questions in advance, and for each keyword in the paragraph, a corresponding question is generated. The extracted keywords can be used as slot-fillers in question generation.

As an example, using the keywords *registration*, *deadline*, *courses* and *later*, as calculated earlier in Table 2, we can associate them with a list of possible question templates such as *When is X?* and *Tell me about X*, where *X* represents a placeholder for a keyword. By creating any permutation of the given questions and keywords, we can produce questions such as: *When is deadline?, When is the courses deadline?, Tell me about courses deadline*. Thus a set of questions related to a particular paragraph is generated based on the lists of questions templates and keywords. The system then maps the list of questions to the corresponding answer (paragraph), and the binding is saved in `PageDataHolder` class.

Slight tuning is necessary to produce correct grammar in the questions (e.g. articles, number agreement; ruling out grammatically impossible permutations, etc.). In the future we plan to use a modification of QASM algorithm (Radev et al., 2001), to grammatical issues into consideration. Especially, in order to build applications for inflected languages like Estonian or Finnish, an algorithm that can take into account the rich morphology of the languages, is necessary.

3.4 Connection to the VH Toolkit

The final step in the system operation is to connect the question data to the Virtual Human Toolkit environment, i.e. to generate the NPC file which contains the set of (user) questions and (virtual human) answers that can be used in the user's interaction with the virtual human. In VH toolkit, the file is edited by the NPCEditor (Leuski and Traum 2011), so the conncection is done by exporting our system data in the same format as is used by the NPCEditor, i.e. via a property list (`plist`). The property list is a markup file format based on the XML standard and often used for data serialization. Unfortunately, the exact specification of the `plist` file structure used in the NPCEditor is not documented, so reverse engineering is forced.

The NPC file description consists of Header, Answers, Questions, and Answers to Questions mapping. Header follows the XML family format defined together with the file encoding, and the `plist` version is defined as well. The format of answers as well as of questions is defined as a list of dictionaries. In both, comments and notes are written in between a number sign. Finally, when answers and questions have been defined, it is possible to proceed to mapping answers to questions. This mapping is also defined as a list of dictionaries, except that the connection between a question and a

corresponding answer is not done via an ID, but via a position in the question/answer list. In addition, the `plist` file also contains other different parameters and settings necessary to make the file working.

A screenshot of the NPCEditor with generated question-answer pairs is shown in Figure 4.

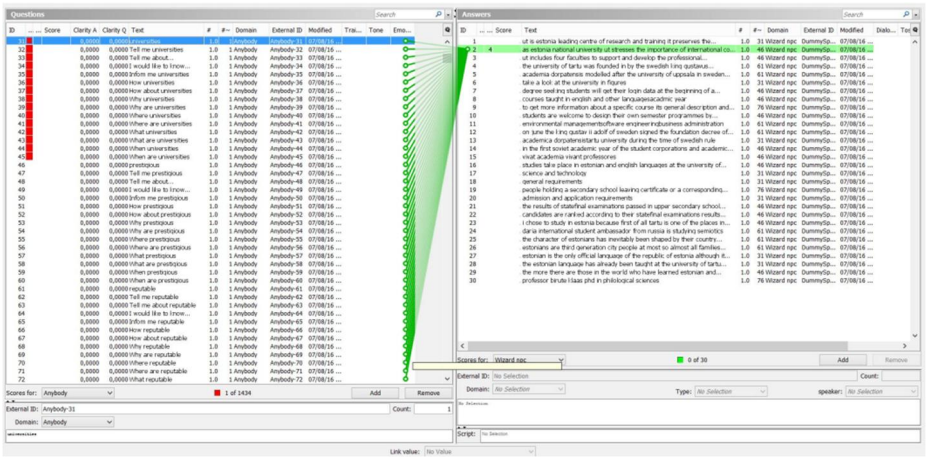

Fig. 4. NPCEditor with generated question-answer pairs

4 Example Dialogue

The preliminary user evaluation of the Virtual Student Office Agent was intended to be completed in August 2016, but it was postponed after the summer till the beginning of the autumn term. In this way, the first evaluation is not just a validation of the system operation but becomes an assessment of the system by real users; the evaluators are expected to be new students who have just arrived at the university and possess real need for various curriculum and course information. Evaluation is planned to be conducted as a web-based questionnaire after each usage situation, with the questions adapted from the evaluation of the WikiTalk (Anastasiou et al. 2013) and from the evaluation of the applications built using the Virtual Human toolkit. Evaluation will concern the functional, operational, and affective features of the Virtual Student Office Agent application: its capability to provide useful and relevant information, its usability as an information providing system (as opposed to going to the Study Office and asking for the information, or looking it up from the website oneself), and the system's acceptability as a reliable agent and companion with which interaction is enjoyable and easy.

A sample dialogue, expected to be appropriately assessed by the potential new students is given in Table 3. U marks the user utterances (the student) whereas S marks the system utterances (Virtual Student Office Agent). Relevant keywords are in italics.

Table 3. An example dialogue with Virtual Student Office Agent

- U: Do I have to pay *school fees* this semester?
- S: Yes, You have to.
- U: How to pay *school fees*?
- S: You can pay it via bank transfer or in cash here in the office.
- U: Can you inform me about *scholarships offers*?
- S: You can have merit-based scholarship, if your average grade is better than B or Student specific scholarship if you are working in some research group.
- U: My grade average is better than B. How do I apply for *merit scholarship*?
- S: You have to send an application by 25th of February. Template can be found on web site of the university.
- U: Thanks
- S: You are welcome

A screenshot of the Virtual Student Office Agent answering a user question is shown in Figure 5. The agent is one of the VH toolkit agents, but the environment is customized to show the University of Tartu main building at the background. The relevant question-answer pair is at the screenshot is:

Student: *I would like to register new subjects, is it possible?*
Agent: *Yes it is possible. You should go to dean's office and inform them.*

Fig. 5. Custom Virtual Human answering a question based on generated questions.

5 Conclusion and future work

The paper has discussed an easy and automated way to create chat-based dialogues from domain information recorded in websites, for the purpose of using the information in virtual agent interactions. The focus has been on the generation of possible user questions in the framework of the Virtual Human Toolkit.

The interaction is assumed to take place between a first year student and a Student Office Agent related to the course and curriculum topics. The dialogue is lively, although more questions and answers are needed for a full coverage of questions related to the possible topics. As for the dialogue capability of the agent, it would be useful to have multiple agendas with larger decision space.

Although human designed dialogues may provide more sophisticated and comprehensive dialogues, the important contribution of the system described in this paper is that it makes dialogue creation easier and automatic. Given the need for creating chat systems and information providing agents to have access to information that already exists in the websites, the current work is a useful step towards dialogue and chat system automation. Moreover, the work distinguishes itself from the previous automated dialogue generation research in that it focuses on the possible user questions rather than building a suitable dialogue policy for the system to execute. In other words, the work attempts to provide mechanisms for anticipating possible user questions based on given information.

Future work will include further development of the various components in the question generation capability, most notably related to the keyword extraction and the actual question formulation. Also more extensive evaluation with real users is planned. User experience with Virtual Humans delivering useful information is useful for the evaluation of the system components and the system performance as a whole, but it is interesting also for studying novel relation between the interacting partners, for building trust and social interaction between the user and the Virtual Agent.

Moreover, since virtual humans allow various communication modalities to be studied, it would be interesting to include various face, gesture, and posture alternatives in the agent's behaviour, and explore their impact on the interaction. Besides the agent's own behaviour and its appropriateness from the user's point of view, it could also be possible for the virtual human to recognize e.g. the user's nodding and gesturing, and be able to adjust its dialogues on the basis of this kind of feedback.

Acknowledgements. The work has been supported by the Estonian Science Foundation project IUT 20-56.

6 References

Anastasiou, D., Jokinen, K., Wilcock, G.: Evaluation of WikiTalk – User Studies of Human-Robot Interaction. In: Kurosu, M. (ed.): Human-Computer Interaction. Interaction Modalities and Techniques, 15th International HCI Conference, Part IV, Lecture Notes in Computer Science, Vol 8007. Springer-Verlag (2013) 32-42 doi: 10.1007/978-3-642-39330-3_4

The British National Corpus, version 3 (BNC XML Edition) 2007. Distributed by Oxford University Computing Services on behalf of the BNC Consortium. URL: http://www.natcorp.ox.ac.uk/

Feng, J., Hakkani-Tur, D., Di Fabbrizio, G., Gilbert, M. and Beutnagel, M.: Webtalk: Towards Automatically Building Spoken Dialog Systems through Mining Websites. Proceedings of the IEEE International Conference on Acoustics Speech and Signal Processing, Toulouse (2006) 1

Gupta, S., Kaiser, G., Neistadt, D. and Grimm, P.: DOM-based content extraction of HTML documents. Proceedings of the 12th International Conference on World Wide Web (2003) 207-214 http://dx.doi.org/10.1145/775152.775182

Hartholt, A., Traum, D., Marsella, S.C, Shapiro, A., Stratou, G., Leuski, A., Morency, J-P., and Gratch, J.: All Together Now: Introducing the Virtual Human Toolkit. Proceedings of the 13th International Conference on Intelligent Virtual Agents (2013)

Jokinen, K. and Wilcock, G.: Multimodal open-domain conversations with the Nao robot. In: Mariani, J., Rosset, S., Garnier-Rizet, M., and Devillers, L. (eds.): Natural Interaction with Robots, Knowbots and Smartphones: Putting Spoken Dialogue Systems into Practice, Springer (2014) 213–224

Jurafsky D.: Text Classification and Naïve Bayes. Stanford University. CS 124: From Languages to Information. Lecture. (2016) https://web.stanford.edu/class/cs124/lec/naivebayes.pdf

Lakshman, P.: Finding keywords using Python. (2016) http://agiliq.com/blog/2009/03/finding-keywords-using-python/

Leuski, A. and Traum, D.: NPCEditor: Creating Virtual Human Dialogue Using Information Retrieval Techniques. AI Magazine 32 Vol 2 (2011) 42–56

Lin, Y., Michel, J-P., Lieberman Aiden, E., Orwant, J., Brockman, W., and Petrov, S.: Syntactic Annotations for the Google Books Ngram Corpus. Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics. Demo Papers, Jeju, Republic of Korea Vol 2 (2012) 169–174

Matsuoka, Y.: Keywords extraction from single document using words co-occurrence statistical information. International Journal on Artificial Intelligence Tools (2003) 157 http://www.worldscientific.com/doi/abs/10.1142/S0218213004001466

McTear, M., Callejas, Z., and Griol, D.: The Conversational Interface. Springer (2016)

Mooney, R.J.: Text Mining with Information Extraction. Multilingualism and Electronic Language Management: Proceedings of the 4th International MIDP Colloquium (2003) 141-160 http://www.cs.utexas.edu/~ml/papers/discotex-melm-03.pdf

Radev, D.R., Qi, H., Zheng, Z., Blair-Goldensohn, S., Zhang, Z., Fan, W., and Prager, J.: Mining the web for answers to natural language questions. Proceedings of the 10th International Conference on Information and Knowledge Management (CIKM '01) (2001) 143-150 DOI: http://dx.doi.org/10.1145/502585.502610

Rahman, A.F.R., Alam, H. and Hartono, R.: Content extraction from HTML documents. Proceedings of the 1st International Workshop on Web Document Analysis (2001) 1-4

Ramos, J.: Using TF-IDF to determine word relevance in document queries. Proceedings of the 1st Instructional Conference on Machine Learning (2003)

Wilcock, G.: WikiTalk: A spoken Wikipedia-based open-domain knowledge access system. Proceedings of the COLING 2012 Workshop on Question Answering for Complex Domains. Mumbai (2012) 57–69