



A procedure for automatic adaptation to the user in intelligent virtual agents

Downloaded from: <https://research.chalmers.se>, 2026-04-13 23:17 UTC

Citation for the original published paper (version of record):

Wahde, M. (2019). A procedure for automatic adaptation to the user in intelligent virtual agents. Multi Conference on Computer Science and Information Systems, MCCSIS 2019 - Proceedings of the International Conferences on Interfaces and Human Computer Interaction 2019, Game and Entertainment Technologies 2019 and Computer Graphics, Visualization, Computer Vision and Image Processing 2019, 2019: 353-357. http://dx.doi.org/10.33965/ihci2019_201906c047

N.B. When citing this work, cite the original published paper.

A PROCEDURE FOR AUTOMATIC ADAPTATION TO THE USER IN INTELLIGENT VIRTUAL AGENTS

Mattias Wahde

*Department of Mechanics and Maritime Sciences, Chalmers University of Technology
412 96 Göteborg, Sweden*

ABSTRACT

This paper describes procedures for improving the functionality of task-oriented intelligent virtual agents (IVAs) and carrying out adaptation to the user in such agents. The functionality improvement is based on automatic extension of input patterns (rules), using a database of interchangeable phrases. This extension results in greater flexibility both regarding the user's input and the agent's output. The adaptation to the user relies on a complexity measure, allowing the agent to assess the level of complexity of the user's input and thus to adapt the complexity level of its response. For the case of a simple travel information agent, the pattern extension procedure resulted in an increase in the number of available patterns of around a factor 20, greatly enhancing the agent's capacity of adapting its output to the user.

KEYWORDS

Task-Oriented Agents, Intelligent Virtual Agents, User Adaptation

1. INTRODUCTION

Intelligent virtual agents (IVAs) are systems that allow a human user to interact with a computer in a natural way, for example using speech and gestures, and receiving speech output in return, often along with an animated three-dimensional on-screen avatar. Such systems are becoming increasingly important in many different application fields including, but not limited to, customer service, automatic booking systems for restaurants, transportation etc., health care and elderly care, travel information systems, and so on (see e.g. Laranjo et al, 2018; Chen and Cheng, 2010). However, an important aspect of such systems which is sometimes overlooked is their ability to adapt their (spoken) output to the user. At a first glance, this might seem like a minor point, but the manner in which a person is addressed, whether by an agent or a human, does in fact have a large impact on the person's perception of the interaction: Some users may prefer an agent that gives very brief, factual answers, whereas others may prefer more verbose and affective speech. Moreover, the suitable manner of speech may also depend on the user's age. That is, if two users are, say, 8 and 88 years old, respectively, an IVA should ideally be able to address them in different ways, focusing on simplicity when interacting with the 8-year old and clarity when interacting with the 88-year old user.

Broadly speaking, IVAs can be divided into two main categories, task-oriented agents and chatbots (Jurafsky and Martin, 2018), where the former are intended to interact with users on a specific task (such as, for example, a reservation request), giving accurate, reliable, and relevant information to the user, whereas chatbots are focused on maintaining a discussion over a large range of topics, but in a more vague and non-committal manner. A currently popular approach to IVAs is to build end-to-end systems using neural networks and deep learning (see e.g. Wen et al, 2016). However, due to their black-box nature, such systems are very difficult to analyze and to modify in a controlled fashion, for example in order to achieve user adaptation as described above. On the other hand, IVAs with more traditional approaches often rely on the identification and interpretation of dialogue acts that classify a speaker's intent into a finite set of classes (Stolcke et al, 2000), and is used when parsing the user's input and determining the appropriate response. While there has been a considerable amount of work on automatic recognition of dialogue acts, the accuracy of that process is around 77% at best (Mezza et al, 2018) over a large benchmark corpus (the Switchboard dialogue act corpus, SWDA), even though higher accuracy has been achieved on smaller data sets or using fewer classes than the 42 used by Stolcke et al (2000). In addition to the limited recognition accuracy for

dialogue acts, it is also not clear how to achieve controlled adaptation to the user in an IVA based on dialogue acts.

A simpler approach is to use a set of specific input patterns or rules (typically hand-crafted) for matching user input, coupled with a set of output patterns for generating the agent's output. Such approaches were used in many early IVAs but have fallen out of fashion, in favor of IVAs based on the two approaches described above. Indeed, it is hard to specify input patterns in a sufficiently general way to handle the great variability in possible user inputs (i.e. the many ways in which a given query can be phrased). However, starting from a rather small set of hand-crafted rules, a procedure (introduced in this paper) can be devised for *automatically* extending the set of allowed inputs, as well as the set of possible outputs, using dictionary information. Applying this procedure, one can thus obtain an IVA that can both handle input and output variability *and* dynamically adapt its speech complexity to that of the user, with the help of a complexity measure for the user input, also defined below. The purpose of this short paper is twofold, namely (1) to introduce a procedure for extending a limited set of hand-crafted rules to a more complete set (see Section 3.1) and (2) to describe a method for adapting an agent's output to the user's input (see Section 3.2), in both cases using the framework of the DAISY dialogue manager, which will now be described briefly.

2. THE DAISY DIALOGUE MANAGER

In this work, a dialogue manager (DM) for task-oriented agents known as DAISY (an abbreviation of *Dialogue Architecture for Intelligent Systems*) has been used. DAISY operates in three stages: First, the system processes the user's input (either speech or text). DAISY allows several methods for input processing (including, for example, the use of dialogue act recognition). However, for the purpose of this study, and motivated by the considerations in the previous section, a rule-based approach has been used, where the user's input is matched to a set of input patterns (exemplified below). Next, if the user's input matches any of the allowed input patterns, cognitive processing takes place, in the form of a sequence of elementary actions that make use of data in the agent's long-term memory (LTM), working memory (WM), or both. Finally, the agent formulates the output using patterns of a similar kind. A typical user-agent interaction thus consists of a triplet I-C-O, where I represents the input processing, C the cognitive processing, and O the presentation of the agent's output.

Note that the patterns need not be rigid, neither for input nor for output: They can (and often do) involve so called query tags for the variable parts of the user's input. As a simple example, a travel information agent may have an input pattern (one among many) of the form "*When is the next bus to <Q1>?*" where the contents of <Q1> would be compared to a set of bus stops, providing a match if the name of the bus stop is available in the agent's database. In DAISY, patterns can be condensed, allowing several options to be represented in a compact fashion. For example, the condensed pattern "[*Can, Would*] you [*help, assist*] me, please?" is a short-hand representation of four specific patterns ("*Can you help me, please?*", "*Would you help me, please?*" etc.).

An agent based on DAISY also maintains information about the context of conversation and can thus, for example, interrupt an ongoing dialogue, answer an unrelated question, and then return to the original dialogue. Furthermore, DAISY features *inference* from context, allowing an agent to respond to questions that require knowledge of the current (or previous) context of conversation. A full description of DAISY will not be given in this short paper; see instead Wahde (2019).

3. ADAPTATION TO THE USER

The procedure for the agent's adaptation to the user takes place in two stages. First, there is an offline stage in which the patterns (both for user input and agent output) are extended to maximize the capability of the agent to understand the user's input and give it opportunity for user adaptation, something that requires that the agent should be able to formulate its output, in a given situation, in several different ways. Second, during interaction with the user, the agent measures the complexity of the user's input, and attempts to adapt its output accordingly.

3.1 Pattern Extension

For this stage, it is assumed that a developer has first generated a rather small set of hand-crafted input patterns, which will act as a starting point. Typically, with the authoring tool that has been implemented for DAISY, such a specification can be completed quite rapidly. For the next stage, i.e. extending the set of input patterns to handle the great variability in user input, the original intention was to use a synonym dictionary on a word-by-word basis. However, even though it is easy to take into account word classes (for example, distinguishing the verb “present” as in “present a paper” from the same noun, meaning “gift”), many words have, of course, several meanings within the same word class, thus making it difficult to automatically extend patterns using synonyms (Ferret, 2010). Thus, the automatic pattern extension procedure was instead implemented on the level of synonymous *phrases* that typically consist of more than one word.

In order to achieve maximum accuracy, crucial for a task-oriented agent, the procedure is based on a new pre-defined database of interchangeable (i.e. synonymous) phrases¹, rather than making use of a corpus-based automatic approach as was done by Manishina et al (2016) and Pavlick et al (2015). Moreover, the dictionary features *dynamic phrases*, i.e. phrases containing unspecified words. For example, the phrase “*the next <Q> departs*” is synonymous with “*the next <Q> leaves*”, regardless of whether <Q> represents (for example) “train”, “flight”, “bus”, or “tram”. Such constructs provide higher accuracy than just (in this example) replacing the word “*departs*” by “*leaves*”, without considering the context.

The construction of the phrase synonym database involves a considerable effort, but on the other hand it can be done once and for all. In order to make sure that the phrases are relevant for human-agent dialogue, a pre-processing method can be applied by which the words in the phrases are checked against a large corpus of spoken dialogue, retaining only phrases containing words that appear frequently in speech. A simple example of a pattern extension is as follows: Starting from the original pattern “*When would you like to go?*”, and using phrase substitutions such as “*when would you => at what time would you*”, “*like to go => wish to go*” etc., an extended pattern was generated that, in condensed form (see also Section 2 above), takes the shape “[*When, At what time*] [*would, do*] you [*like, wish, want, prefer*] to [*go, depart, travel*]?”.

3.2 Complexity Matching

In order to adapt to the user’s way of speaking, the agent must somehow measure, or at least estimate, the level of complexity of the user’s speech. A coarse-grained option for doing so would be to equip the agent with a camera, allowing it to assess the approximate age of the user. While possible as a *complement* to the approach described below, camera-based user age estimation would only provide partial information, at best. Thus, a more reliable approach would be to infer the level of complexity of the user’s input.

Several measures of speech complexity (or quite often the reverse, i.e. speech simplicity) exist, one of the most well-known, and thoroughly validated, being the Flesch-Kincaid (F-K) readability measure (Kincaid et al, 1975). However, this measure is primarily intended for analyzing a body of text spanning many sentences, and can give inaccurate or at least counter-intuitive results when applied to single sentences. Moreover, the F-K measure is not designed specifically for use with *spoken* input, where sentences are often very short, sometimes just a single word. Thus, while the F-K measure is indeed implemented as an alternative in DAISY, here a much simpler complexity measure has been used, namely the number of syllables in the user’s input. One could also have used the number of words, but this measure was deemed too primitive and coarse-grained (keeping in mind, again, that the user’s input often consists of only a few words). With the chosen syllable-based measure, a better estimate is obtained. For example, compare the two semantically equivalent user-input sentences (in response to a statement by the agent regarding, say, a museum): (I) “*Can you show an image of it?*” (8 syllables), and (II) “*Can you please display a visual representation of the museum?*” (19 syllables). Even though the second sentence is perhaps a bit too contrived to be a likely user input, it illustrates the complexity measure quite well, when contrasted with the first sentence.

The complexity matching procedure operates as follows: For a given user input, the agent registers the level of complexity (denoted *c*) using the measure just described. Next, if a matching pattern is found, the agent computes its relative complexity, by first sorting the complexity values for *all* versions of the matching

¹ The database is available at <http://www.me.chalmers.se/~mwahde/DoIP/doipDescription.html>

pattern (expanding the condensed pattern as illustrated in Section 2) in ascending order, and then computing the rank of the user input.

For example, if a condensed pattern can be expanded to 48 different specific patterns (which is the case for the condensed pattern shown at the end of Subsection 3.1), and the *actual* user input would be ranked (for example) as the 39th, the relative complexity r is computed as $r = 39/48 = 0.8125$. This relative measure is used instead of the absolute complexity value c , since there is no reason to believe that the agent's selection of possible outputs would cover the same complexity range as the inputs. Instead, when responding to the user (after processing the input), the agent would repeat the procedure, ranking the complexity values of the possible *output* patterns, and selecting (for output) the pattern whose relative complexity is closest to that of the user input. Continuing the example, if the condensed output pattern (not shown here) allows for 16 different specific patterns, ranked in ascending order of complexity, the output would in this example be chosen as the pattern with index $16 \times 0.8125 = 13$. Of course, if the index thus computed is not an integer, the index should be set as the nearest integer.

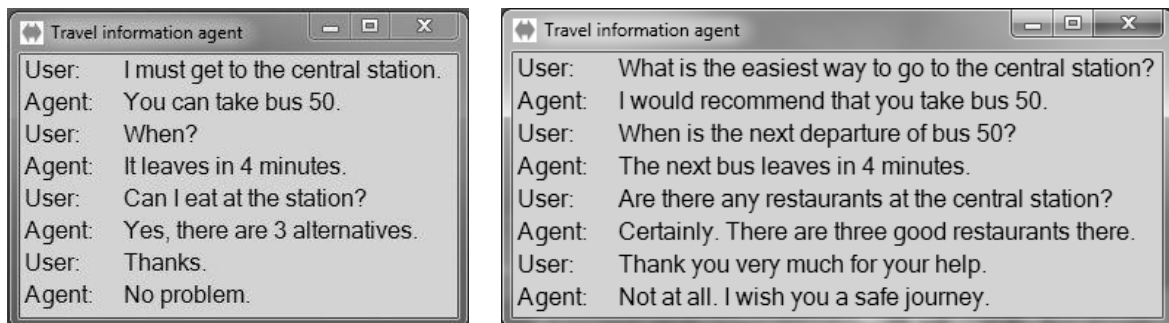


Figure 1. Left panel: A dialogue between an agent and a user who is in a hurry, and therefore expresses requests in a very brief manner. Right panel: A dialogue between *the same* agent and a more verbose and relaxed user. As can be seen in the dialogues, the agent adapts its output to the user's input in both cases, using the procedure described in Subsection 3.2

4. PRELIMINARY RESULTS AND DISCUSSION

The methods for pattern extension and complexity matching described above have been implemented and are currently undergoing testing. In order to test the pattern extension method, a travel information agent was implemented with the capability of giving information of the kind exemplified in Figure 1. The phrase synonym dictionary is not yet complete (i.e. covering phrases useful in *any* topic). Instead, in its current form, it contains a set of basic phrases that are relevant in any context and a set of phrases specifically relevant for a travel information system of the kind exemplified in the figure.

For the sample dialogue in Figure 1, the hand-crafted agent (i.e. the starting point) contained a total of 7 input patterns and 7 output patterns, distributed over the various user-agent interactions defined in the dialogue. Next, the pattern extension procedure was applied, resulting in a total of 118 input patterns and 165 output patterns, representing a combined increase by a factor of 20.2. The magnitude of the increase in the number of patterns depends on several factors, but the numbers just given provide an indication of the typical magnitude. Moreover, in this agent, rather than rigidly selecting the output pattern index as described in Section 3.2 above, a modification was implemented such that the output pattern is selected from a range of indices centered on the computed index, to allow also greater *variability* in the agent's output, which is an important factor to consider in task-oriented agents.

The initial results regarding adaptation to the user, which are qualitative in nature, show that the agent does indeed generate output that matches the level of complexity of the user's input rather well. These initial tests will be followed by a more rigorous examination, where a large group of users will be exposed to agents either with or without user adaptation, and will then give (subjective, but detailed) feedback on the user experience for the two cases. Another rather natural observation is the fact that the complexity matching works well only if the agent has a large set of specific patterns (both for input and for output) to choose from, thus making the first step (pattern extension) a crucial component of the system.

5. CONCLUSION

A procedure for automatic extension of patterns (rules) for intelligent virtual agents has been introduced, allowing greater variability in the user's input and also making it possible for an agent to adapt the complexity level of its output to that of the user, provided that a database of synonymous phrases is available. Such a database is currently under construction. Once it has been completed, it should prove useful in many different task-oriented agents. A specific method for adaptation to the user's style of speech in task-oriented agents has also been introduced and briefly described. Preliminary results, for the case of a simple travel information agent, indicate that the pattern extension method leads to a large increase in the set of patterns, and that the method for adaptation to the user achieves the intended objective of automatic matching to the user's style of speech.

REFERENCES

- Chen, B. and Cheng, H. H., 2010. A review of the applications of agent technology in traffic and transportation systems, *IEEE Transactions on Intelligent Transportation Systems*, Vol. 11, No. 2, pp. 485-497.
- Ferret, O., 2010. Testing semantic similarity measures for extracting synonyms from a corpus. *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC'10)*, Malta, pp. 3338-3343.
- Jurafsky, D and Martin, J.H. 2018. *Speech and Language Processing* (3rd ed.), Prentice-Hall, New Jersey.
- Kincaid, J.P. et al 1975. Derivation of new readability formulas (automated readability index, fog count, and Flesch reading ease formula) for Navy enlisted personnel. *Research Branch Report 8-75. Chief of Naval Technical Training.*
- Laranjo, L. et al 2018. Conversational agents in healthcare: A systematic review. *Journal of the American Medical Informatics Association*, Vol. 2, No. 9, pp. 1248-1258.
- Manishina, E. et al 2016. Automatic corpus extension for data-driven natural language generation. *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, Paris, France, pp. 3624-3631.
- Mezza, S. et al 2018. ISO-standard domain-independent dialogue act tagging for conversational agents. *Proceedings of the 27th International Conference on Computational Linguistics*, Santa Fe, NM, USA, pp. 3539-3551
- Pavlick, E. et al 2015, PPDB 2.0: Better paraphrase ranking, fine-grained entailment relations, word embeddings, and style classification. *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing*, Beijing, China, Vol. 2, pp. 425-430
- Stolcke A. et al 2000 Dialogue act modeling for automatic tagging and recognition of conversational speech. *Computational Linguistics*, Vol. 26, No. 3, pp. 339-373
- Wahde, M. 2019. A dialogue manager for task-oriented agents based on dialogue building-blocks and generic cognitive processing. *Proceedings of the 9th Workshop on Applications of Software Agents* (in press)
- Wen, T.-H. et al 2016. A network-based end-to-end trainable task-oriented dialogue system. *arXiv preprint arXiv:1604.04562*.