AUTOMATIC QUESTION GENERATION FOR VIRTUAL HUMANS

Evania Lina Fasya

Master of Science

Human Media Interaction

Graduation committee:

dr. Mariët Theune (1st supervisor)

dr.ir. Rieks op den Akker (2nd supervisor)

August 2017

University of Twente

Enschede, The Netherlands

ABSTRACT

Alice, a virtual human that is created based on the ARIA-VALUSPA framework, is a representation of the main character from a classic novel Alice's Adventures in Wonderland. Alice needs the domain knowledge of the Alice in Wonderland story in order to talk about the story with its users. However, the current domain knowledge of Alice is still created manually, and it can be difficult to create more virtual humans in other domains or to extend the knowledge of Alice.

This research aims to prepare the domain knowledge of Alice in a more automated process by developing an automatic question generation system. The system is called Alice Question Generation (AQG) and it makes use of two semantic tasks; Semantic Role Labeling (SRL) and Stanford Dependency. The main task of the AQG system is to generate questions and answers (QAs) about Alice in Wonderland. The generated QAs will be stored in the QAMatcher, which is a tool that stores the domain knowledge of Alice in a QA pair format. The QAMatcher works by matching a user's question with a number of prepared questions using text processing algorithms, and then gives the answer that is linked to the matched question.

The first phase in developing the AQG system is observing the SRL and Dependency patterns. The second phase is creating the QA templates. These templates were evaluated twice, with error analysis and improvements conducted after each evaluation. Next, a user study using the QAMatcher was conducted. The user study result shows that the current AQG system cannot be used by itself in a virtual human. More varied questions that ask about the same thing are necessary to enable the QAMatcher to match the user's questions better. This research discusses the important aspects when implementing the automatic question generation for virtual humans at the end of the report.

ACKNOWLEDGMENTS

The author would like to thank dr. Mariët Theune for all the reviews and feedbacks that enable the thoughtful and critical discussion from the research topic until the final project; dr.ir. Rieks op den Akker for the feedback on the final project and the inspiration about natural language processing; and Jelte van Waterschoot for the update on ARIA-VALUSPA project and the discussion about retrieving information from a narrative.

The author would also like to thank the Ministry of Communication and Informatics of Indonesia for granting a scholarship in Human Media Interaction at the University of Twente and giving the chance of pursuing the master education based on the author's passion and competence.

Finally, this final project would not be possible without the support from the family and friends. The author would like to thank her mother for all the love; her father for the inspiration; two sisters for the fun and support; Niek for the encouragement and comfort; all the housemates for the friendship; and all other family members and friends.

TABLE OF CONTENTS

Page

Page

1. INTRODUCTION

ARIA-VALUSPA, an abbreviation for the Artificial Retrieval of Information Assistants Virtual Humans with Linguistic Understanding, Social skills, and Personalized Aspects, is a project of the Horizon 2020 research programme of the European Union. The project intends to create a framework of virtual humans which are capable of conducting multimodal interaction with their users in challenging situations, such as facing an interruption, or reacting appropriately according to emotion and gesture changes. One virtual human that is being developed is called Alice, representing the main character of the classic novel written by Lewis Carroll, Alice's Adventures in Wonderland. There are several work packages that are involved in the ARIA-VALUSPA project. But the specific work package that is being carried out at the University of Twente is called Multi-Modal Dialogue Management for Information Retrieval.

There are some challenges in developing multi-modal dialogue management for information retrieval. One of them is preparing the domain knowledge for the virtual human. As the representation of the character Alice in the story of Alice in Wonderland, the virtual human - Alice - needs to have the domain knowledge of the story. However, the current domain knowledge for Alice is still created manually, and it can be difficult to create more virtual humans in other domains or to extend the knowledge of Alice (e.g. extending the knowledge from only knowing the story of the novel into knowing the story of the writer).

This research aims to prepare the domain knowledge of Alice in a more automated process by using an Automatic Question Generation approach. Automatic question generation is an activity that takes a text resource as an input and generates possible questions (and answers) that can be asked from the resource. The generated questions and answers are furthermore stored in the QAMatcher, which is a tool that manages the domain knowledge of Alice. The QAMatcher works by matching a user's question with a number of prepared questions using text processing algorithms, and then gives the answer that is linked to the matched question.

There are two other approaches that were considered to prepare the knowledge of Alice. The first one is collecting question and answer pairs from the internet. The benefit of this approach is that the questions from the internet are usually asked by real people. Implementing this approach allows Alice to have some insights of what kind of Alice-in-Wonderland-questions do people in general are curious about. The second approach is question answering. Question answering lets the virtual human search the answer of a question directly in a resource that is made available through a prepared "knowledge base" [1].

The automatic question generation approach is finally chosen because the developing time is reasonable compared to question answering approach. In addition to that, it can be easily adapted for other virtual humans in other domains, compared to collecting question and answer pairs from the internet which require more manual process.

As a virtual human that is based on the ARIA-VALUSPA framework, Alice is expected to be able to respond accordingly to the users in challenging situations, such as asking for a confirmation when Alice could not hear the user well. This research, however, only explores the domain knowledge of Alice, which is the story of Alice in Wonderland. Therefore, the other conversation elements such as handling interruptions, greetings, etc., are not the focus of this research.

In the next chapter, the concept of conversational agents is explained, followed by its relation with virtual humans. In chapter 3, the current implementation of the ARIA-VALUSPA is described. In chapter 4, question generation is described. Chapter 5 describes the creation of a question generation system for Alice. Chapter 6 explains the initial evaluation and the improvement for the system. Chapter 7 explains the next evaluation that was conducted by 6 annotators. Chapter 8 describes a user study using the QAMatcher. Finally, chapter 9 presents the conclusions and discusses future work.

2. CONVERSATIONAL AGENTS

A conversational agent is a system that can communicate with its users by understanding spoken or textual language. Most conversational agents in the beginning of 2000s, however, are intended to communicate through speech rather than text, and so they are also known as spoken dialogue system [2]. Similar with spoken dialogue systems, virtual humans are also a type of conversational agents. Virtual humans are able to carry a conversation with their users through speech like spoken dialogue systems. However, a noticeable difference of spoken dialogue systems and virtual humans is that virtual humans have visual representations. These visualizations are expected to be able to generate nonverbal behaviors just like real humans.

Dialogue systems and virtual humans are described in more detail in section 2.1 and section 2.2 below. Furthermore, a specific component of conversational agents, dialogue manager, is described separately in section 2.3 because the dialogue manager component is related with the focus of this research.

2.1 Dialogue Systems

A dialogue system is a computer system that is able to have a conversation with humans. One implementation of dialogue systems is spoken dialogue systems used in commercial applications such as travel arrangement system and call routing. How May I Help You [3] is an example of a spoken dialogue system whose task is automatically routing telephone calls based on a user's spoken response to the question "How may I help you?". Figure 2.1 shows an example of a conversation between a user and the How May I Help You (HMIHY) system [3].

There are several activities behind a spoken dialogue system in order to understand what the users say and give back appropriate responses. Typically, these activities are managed within several components. An illustration of the components of a typical spoken dialogue system [2] is shown in Figure 2.2.

System : How may I help you?

User : Can you tell me how much it is to Tokyo?

System : You want to know the cost of a call?

User : Yes, that's right.

System : Please hold on for rate information.

Fig. 2.1.: A conversation between a user and the HMIHY system [3]

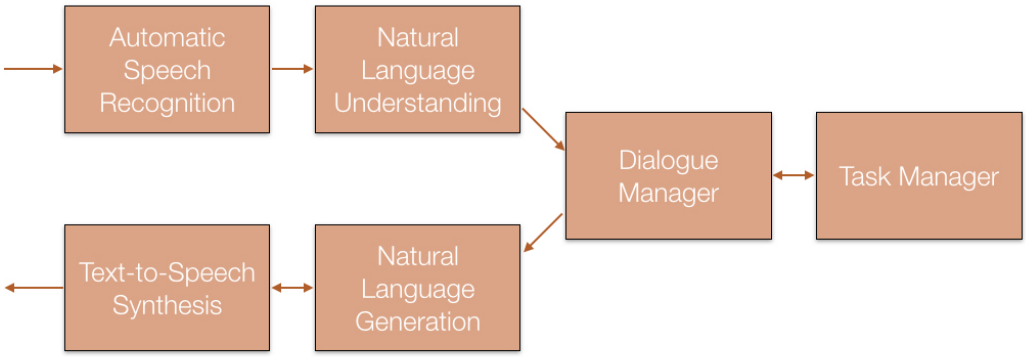

Fig. 2.2.: An architecture of the components of a spoken dialogue system [2]

The Automatic Speech Recognition (ASR) component takes the audio input from the user through a desktop microphone or a telephone, and then returns a transcribed string of words to the Natural Language Understanding (NLU) component. The NLU's task is to produce the semantic representation of the strings from the ASR. The Dialogue Manager processes the semantic representation from the NLU and produces the most appropriate response for the Natural Language Generation. The Dialogue Manager manages all the dialogues with the help from the Task Manager. The Task Manager consists of the current communication goals (e.g. the user wants to find direct flights on Thursday, the system wants to give the information about some available flight schedules). The Natural Language Generation (NLG) module gets the output from the dialogue manager and decides how to say this output to the user in words. The Text-to-Speech component gives these words a waveform so that the words can be produced as a speech.

2.2 Virtual Humans

Virtual humans are different from spoken dialogue systems because virtual humans have visualizations, such as a body or a face. Beside of that, virtual humans that are created based on the ARIA-VALUSPA framework are not only expected to understand the spoken and written language, but also expected to understand nonverbal human behaviors.

Because of their human likeness, virtual humans can be used to train real human's social skills when facing stressful situations by simulating the scenario in a safe virtual world. An example of this implementation is Mission Rehearsal Exercise system [4] which trains the user's leadership skills in a warzone. Virtual humans can also be implemented in museums to increase the interest and engagement of the visitors (e.g. Ada and Grace [5]); or to do interviews with patients for healthcare support (e.g. Ellie [6]).

The architecture of a virtual human is more complex than the typical architecture of spoken dialogue systems because it involves more modules such as nonverbal behavior understanding and nonverbal behavior generation.

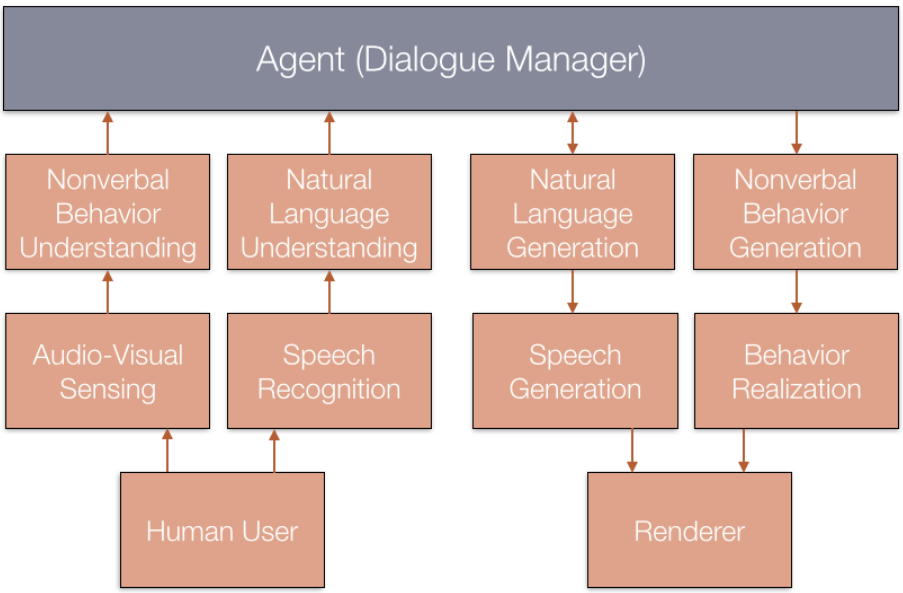

Fig. 2.3.: Virtual Human Architecture [7]

Figure 2.3 shows the common architecture of a virtual human [7]. The architecture is almost similar to the typical architecture of spoken dialogue systems [2]. However, as shown

in figure 2.3, the virtual human architecture also involves Audio-Visual Sensing, Nonverbal Behavior Understanding, Nonverbal Behavior Generation, and Behavior Realization.

When a human user talks to the virtual human, his speech is transformed into a textual representation by the Speech Recognition module. The text is then translated into semantic representation by the Natural Language Understanding module. This process is similar to the spoken dialogue system's process except that the human user's expression and nonverbal communication are also recognized by the Audio-Visual Sensing module in the virtual human. The Nonverbal Behavior Understanding module takes the information from the Audio-Visual Sensing module and links certain observations to higher-level nonverbal communicative behaviors (e.g. attention value, head position). Based on the nonverbal communicative behavior values and the semantic representation of the speech, the Dialogue Manager replies back with the most appropriate response. The Dialogue Manager, which is labeled as the Agent in [7], manages all the dialogues, similar to the Dialogue Manager module in the spoken dialogue system architecture 2.2. The responses from the dialogue manager are sent to the Natural Language Generation and Nonverbal Behavior Generation so that they can generate the appropriate response using speech and behavior. The response can be produced by the Speech Generation module using text-to-speech or pre-recorded audio. The Behavior Realization module synchronizes all behaviors such as speech, gestures, and facial expressions, and gives them for a renderer to show.

An example of a virtual human framework is Virtual Human Toolkit (VHToolkit) [7] which main focus is to create a flexible framework that allows the creation of different kinds of virtual humans. Another example is SEMAINE [8] which main goal is to create virtual listeners that are able to engage in a conversation with a human user in the most natural way. Each module in the architecture of VHToolkit or SEMAINE can consist of one or more tools. For example, VHToolkit uses one tool that handles the Audio-Visual Sensing and Nonverbal Behavioral Understanding, while SEMAINE uses three separate tools in these two modules. The details of these modules and the rest of the modules in the virtual human architecture are not explained further, except for the Dialogue Manager which is described in the next section.

2.3 Dialogue Management

Dialogue Management is a task which is carried out after the behavior understanding and the natural language understanding tasks. The tasks of a Dialogue Manager are to take the semantic representation of words from the NLU module and the output from the Nonverbal Behavior Understanding module, manage the dialogues, and give back the appropriate response to the verbal/nonverbal generation modules. There are different types of dialogue managers based on the goal of the conversational agents. The common dialogue managers can be separated into four types [2] as follows.

2.3.1 Finite-State

Finite-state is the simplest architecture where the system completely controls the conversation with the user. It asks the user a series of questions, ignoring anything that is not a direct answer to the question and then going on to the next question. For example, the system will always ask the question "What city are you leaving from?" until the system recognizes a city name from the user's response, and then the system continues to the next question. Figure 2.4 illustrates a simple finite-state automation architecture of a dialogue manager in a spoken dialogue system [2].

2.3.2 Form-based

Form-based is more flexible than the finite state dialogue manager. It asks the user questions to fill slots in the form, but allows the user to guide the dialogue by giving information that fills other slots in the form. For example, if the user answers "I want to leave from Amsterdam on February 24th" to the question "What city are you leaving from?", the system will fill in the slots ORIGIN CITY and DEPARTURE DATE. After that, the system can skip a question "Which date do you want to leave?" and move on to a question "Where are you going?". Table 2.1 shows the example of slots and the questions that a form-based dialogue manager can ask.

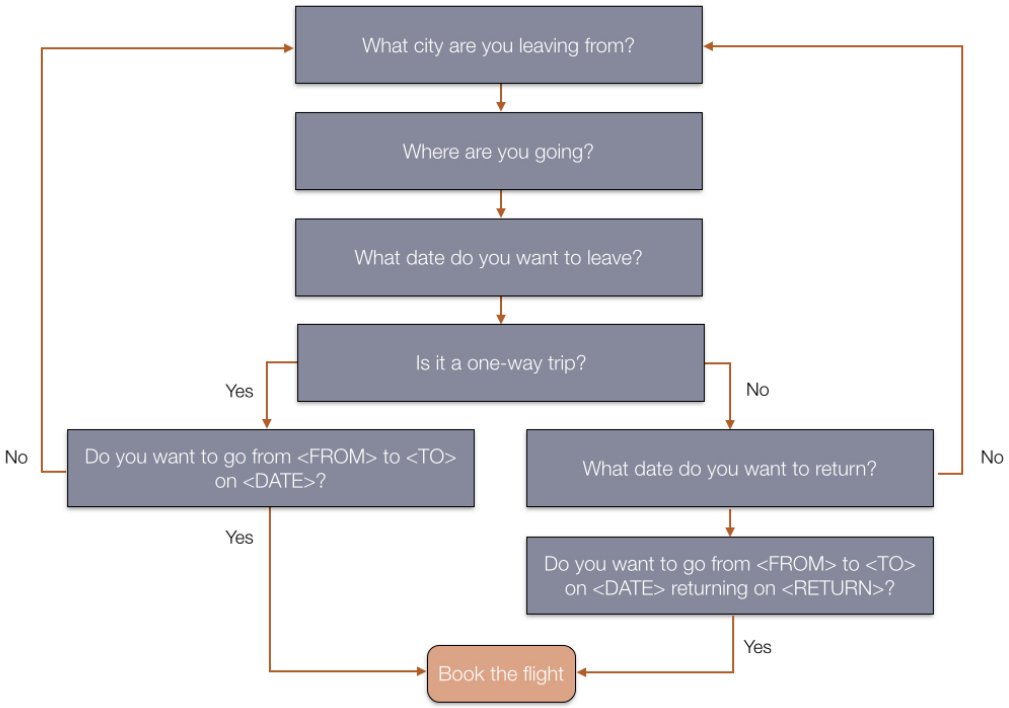

Fig. 2.4.: A simple finite-state automation architecture [2]

Table 2.1.: Example of slots and questions in a form-based dialogue manager

Slot	**Question**
ORIGIN CITY	"What city are you leaving from?"
DEPARTURE DATE	"Which date do you want to leave?"
DESTINATION CITY	"Where are you going?"
ARRIVAL TIME	"When do you want to arrive?"

2.3.3 Information-State

Information-state is a more advanced architecture for a dialogue manager that allows for more components, e.g. interpretation of speech acts or grounding. Different from the finite-state or the form-based architecture which only allow the computer to ask questions,

the information-state architecture is able to decide whether the user has asked a question, made a suggestion, or accepted a suggestion. This architecture thus can be more useful than just form-filling applications that are usually the implementation of the finite-state and form-based architecture. An information-state based dialogue management can assign tags to the dialogues, for example, a response "Hello" can be interpreted as a greeting, thus it can be tagged with the attribute GREET. Another example, a response "There is one flight in the morning at 9.15" can be tagged with the attribute SUGGEST. Table 2.2 illustrates some dialogue acts in an information-state based architecture adapted from [2].

Table 2.2.: Some dialogue acts used in an information-state based dialogue manager called Verbmobil-1

Tag	**Example**
GREET	Hello Ron
INTRODUCE	It's me again
REQUEST-COMMENT	How does that look?
SUGGEST	From thirteenth through seventeenth June
ACCEPT	Saturday sounds fine

2.3.4 Plan-Based

Plan-based dialogue management is also a more sophisticated architecture compared to the finite-state and form-based. The plan-based model allows the system to know the underlying intention of utterances. The model can be further explained using the dialogues in Figure 2.5.

Each of the discourse segment within the discourse in figure 2.5 has a purpose held by the person who initiates it. Each discourse segment purpose (DSP) has two relations called dominance and satisfaction-precedence. When a DSP1 dominates DSP2, it means that satisfying DSP2 is intended to provide part of the satisfaction of DSP1. When a DSP1

U1 I need to travel in May.

S1 And, what day in May do you want to travel?

U2 OK uh I need to be there for a meeting that's from the 12th to the 15th.

S2 And you're flying into what city?

U3 Seattle.

S3 And what time would you like to leave Pittsburgh?

U4 Uh hmm I don't think there's many options for non-stop.

S4 Right. There's three non-stops today.

U5 What are they?

S5 The first one departs from Pittsburgh Airport at 10:00am, arrives at Seattle Airport at 12:05 their time. The second flight departs from Pittsburgh Airport at 5:55pm, arrives at Seattle Airport at 8pm. And the last flight departs from Pittsburgh Airport at 5:55pm, arrives at Seattle Airport at 10:28pm.

U6 OK I'll take the 5ish flight on the night before on the 11th.

S6 On the 11th? OK. Departing at 5:55pm arrives at Seattle Airport at 8pm, U.S. Air flight 115.

U7 OK.

Fig. 2.5.: A discourse example from a telephone conversation between a user (U) and a travel agent system (S)

satisfaction-precedes DSP2, it means that DSP1 must be satisfied before DSP2. Therefore, the structure of the discourse in Figure 2.5 can be summarized in Figure 2.6.

The explanation of Figure 2.6 is as follows:

1. DSP1: Intend U (S finds a flight for U)
2. DSP2: Intend S (U tells S about U's departure date)
3. DSP3: Intend S (U tells S about U's destination city)
4. DSP4: Intend S (U tells S about U's departure time)

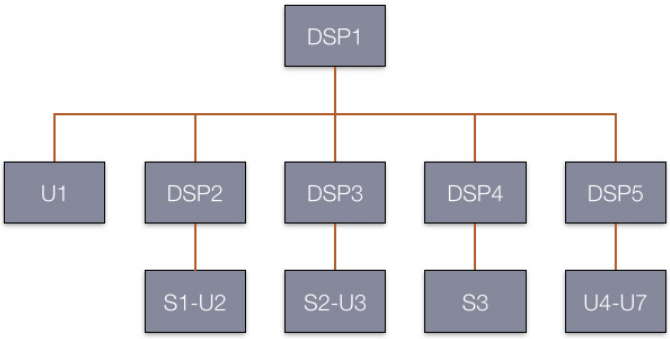

Fig. 2.6.: The discourse structure of the discourse in Figure 2.5.

5. DSP5: Intend U (S finds a nonstop flight for U)

Since DS2 - DS5 are all subordinate to DS1, Figure 2.5 can be reflected in the dominance relationship: DS1 dominates DS2 Λ DS1 dominates DS3 Λ DS1 dominates DS4 Λ DS1 dominates DS5. Moreover, since DS2 and DS3 need to be satisfied before DS5, thus they can be reflected in the satisfaction-precedence relationship: DS2 satisfaction-precedes DS5 Λ DS3 satisfaction-precedes DS5.

As shown in Figure 2.6, a plan-based dialogue management allows the system to understand the intention of a discourse segment. When the system asked “And what time would you like to leave Pittsburgh?”, the user did not answer right away because the user did not know the schedule for direct flights. The system understood this and gave some options of direct flights before continuing the plan of reserving the departure time.

3. ARIA-VALUSPA

ARIA-VALUSPA is a project that intends to develop a framework of virtual humans that allows a robust interaction between a virtual human and a user in the most natural way. As described in the beginning of the introduction, Alice is one virtual human that is developed based on the ARIA-VALUSPA framework. The architecture of Alice is based on the common virtual human architecture described in section 2.2. Alice has an Audio-Visual Sensing and Speech Recognition module, as well as the Nonverbal Behavior Understanding and Natural Language Understanding. Alice also has the Natural Language Generation, Speech Generation, Nonverbal Behavior Generation, and the Behavior Realization. The focus on each module is to create the most natural interaction as possible by considering some common elements in a conversation such as facial expressions of emotions, gestures, interruption, etc.

The focus of this research topic is, however, the knowledge of Alice - which is more related to the Dialogue Manager in the architecture. In section 3.1, the current state of the Alice's Dialogue Manager is described. Furthermore, an overview of Alice's domain knowledge is discussed in section 3.2.

3.1 The Dialogue Manager of Alice

Alice is developed using the information-state based architecture dialogue manager [9]. As described in section 2.3, an information-state based architecture allows Alice to interpret the intent of the utterance. For example, when a user asks "What do you think of the Mad Hatter?", Alice categorizes this utterance as intent "setQuestion". Alice assigns an intent based on some rules (e.g. assign setQuestion intent if the utterance consists of the word "think","Mad", and "Hatter"). By having these categories, Alice can respond appropriately to an utterance by an intent "inform", for example.

The specific dialogue manager that is used is called Flipper [10]. Flipper allows Alice to have a flexible set of templates that can specify what kind of behavior to perform at a state.

These templates are called FML templates [9]. When a response has been decided, Flipper sends a the response to the Behavioral Generation. Besides the nonverbal behavior handling, an extension of Flipper has been developed to enable Alice to handle dialogues. The dialogue handling and the nonverbal behavior handling can be processed simultaneously. The complete overview of Alice's dialogue manager is shown in Figure 3.1.

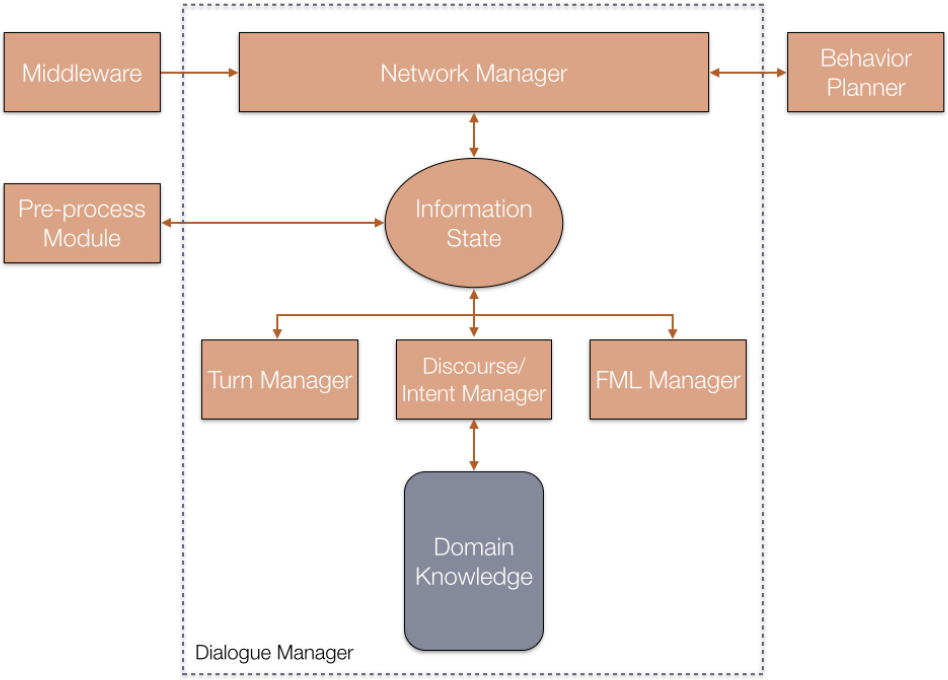

Fig. 3.1.: The overview of Alice's Dialogue Manager [9]

The scope of the Dialogue Manager is marked with the dashed outline. It takes the output from middleware, such as the output from Social Signal Interpretation (SSI) module [11] that is used by Alice to understand the user's behavior. The Dialogue Manager also sends a user utterance to the Pre-Processing Module and takes the output which consists of the intent of an utterance, such as "setQuestion".

Within the scope of the Dialogue Manager, the Network Manager is responsible to manage the current state of Flipper. Some examples of the states are getting the input from the SSI and integrating the streams to the Information State, or sending a response from the

Information State to the Behavioral Planner, as well as receiving feedback of whether the response has been delivered successfully to the user. The Turn Manager module manages the turns in the dialogue. For example, when the user speaks the turn is marked as "user" while when Alice speaks, the turn is marked as "Alice". The system also notices when the user has been silent for a while, then the turn will be changed to Alice. The Discourse/Intent Manager takes the intent from a user's utterance and return an appropriate agent's intent. The discourse part specifies the phase of the discourse, such as opening phase, information retrieval phase, or closing phase. The FML Manager decides the most appropriate FML template from the agent's intent that has been returned by Discourse/Intent Manager module. FML template consists of parameters such as subjects, objects, or emotions. Finally, the Domain Knowledge is retrieved by the Discourse/Intent Manager based on the current intent. For example, when the intent is asking an information about the white rabbit, the returned information from the Domain Knowledge is "The white rabbit is a strange rabbit with a watch inside his waistcoat-pocket".

3.2 The Domain Knowledge of Alice

The domain knowledge of Alice is stored in a system called QAMatcher and is formed in a question and answer pair format. When a user asks a question to Alice, the QAMatcher matches the user's question with a list of questions by using a text processing algorithm. When a matched question has been found, the answer to the matched question is returned back to the user. The question and answer pairs are prepared before-hand and are stored in the QAMatcher's resource directory. Automatic question generation is the approach that is used to prepare these question and answer pairs in the QAMatcher.

There are two types of knowledge that Alice can have, they are the knowledge about Alice in Wonderland story and the knowledge about general conversation, e.g. greeting, inform, etc. These types are called domain-dependent and domain-independent according to Dynamic Interpretation Theory (DIT++) taxonomy of communicative function [12]. The focus of this research, however, is the domain-dependent knowledge, which is the knowledge about Alice in Wonderland story.

4. QUESTION GENERATION

Automatic question generation, or more simply known as question generation, is an activity that takes a text resource as an input and generates possible questions (and answers) that can be asked from the resource. This approach allows the generation of the questions and answers that can be used in the QAMatcher.

Recent research shows that there are several applications of a question generation system, such as education, social media security, and conversational agent. These applications are explained in more detail in section 4.1. Despite the application of question generation systems, a question generation system can be developed using several approaches. The common approaches are explained in section 4.2. The discussion of the implementation of a question generation system and what approach can it be developed for Alice is provided in section 4.3.

4.1 Implementation of Question Generation

Many question generation (QG) systems are used in educational applications, such as skill development assessment and knowledge assessment [13]. G-Asks is an example of QG implementation in skill development assessments [14]. G-Asks generates trigger questions that can support students to learn through writing. For example, students are encouraged to learn varied opinions from other research. However, when a student cite an opinion from other research in his own writing, a new follow-up question can be formed from this citation, such as "Which statements of the other research that form this opinion?". G-Asks is able to generate this "evidence support" type of question to support the academic writing.

A QG system that is developed for knowledge assessment was conducted by Heilman and Smith [15] [16] [17]. Heilman and Smith created this QG system with the goal of helping teachers in creating exam and quiz materials. A user study was conducted with real teachers and the result was the tool indeed helped teachers to prepare the question and answer pairs faster with less effort [18].

Another QG system that is developed for knowledge assessment was conducted by Mazidi and Nielsen [19]. They managed to construct deeper questions than factoid questions and outperformed the result from Heilman and Smith.

Besides the common applications of QG in educational applications, QG can also be used in the social media security domain. For example, getting personal information from a user's social media account, and generate questions from it [20]. The questions then are asked back to the user for authentication when a user forgets his password.

A research of QG that is done in a conversational agent domain was conducted by Yao et al. [1]. They used two QG tools to create question and answer pairs to be used as the knowledge base for a conversational character that can communicate with real humans. They used 14 Wikipedia articles as the topic and the question and answer pairs that have been generated from the tools are then stored in question and answer matching tool called NPCEditor [21]. The first QG tool that they used is the QG system that was developed by Heilman and Smith [15]. The second tool that they used is called OpenAryhpe which was developed by Yao et al. themselves based on a Question Answering framework called OpenEphyra [22]. The difference between OpenAryhpe and the Question Transducer is that OpenAryhpe expands some components so that the tool can recognize new synonyms and is able to recognize time, distance, and measurement more precisely.

Yao et al. concluded that the question and answer pairs that were generated by both QG tools can be used as the knowledge base for a conversational character [1]. However, there are some problems that they faced. First, there are some mismatches between the actual questions that the users ask and the generated questions. This happens because question generation tools only provide questions which have the answers available in the source text. Based on this problem, they planned to use the sample questions from the user study to analyze the frequent questions that the users ask for future research. The second problem is that there is a gap between the vocabularies used by the users with the generated questions. Based on this problem, they planned to use other lexical resources to provide synonyms for the words in the future research.

4.2 Approaches in Question Generation

The recent approaches in question generation (QG) are varied based on the Natural Language Processing (NLP) tools available to the researchers [23]. However, the direction of the approaches can be classified into two categories, syntactic or semantic [19]. Syntactic approach explores the use of syntactic tools such as Stanford Parser and Tregex and uses them as the foundation of its QG system. On the other hand, the semantic approach explores the semantic tools such as Stanford Dependency and Semantic Role Labels (SRL) as the foundation of its QG system. Either approach that is implemented as the foundation of the QG system, however, does not limit the system to make use the opposite approach. For example, a QG system that uses syntactic tools as its foundation can still make use of semantic tools to make the QG system perform better. The syntactic and the semantic approaches are explained in more detail in this section using two prior research from Heilman and Smith, and Mazidi and Nielsen.

4.2.1 Heilman and Smith

The QG research of Heilman and Smith [15] [16] [17] can represent the syntactic approach. There are several syntactic tools that Heilman and Smith used for their QG system. For example, they used Stanford Phrase Structure Parser to automatically sentence-split, tokenize, and parse input texts resulting in a Penn Treebank structure (e.g. Alice = NNP, watched = VBD, the = DT, white = NNP, rabbit = NNP). They also used the Tregex tree searching language to identify the syntactic elements of the sentence (e.g. subject and object of the sentence). They used Supersense Tagger to generate the answer phrase mainly for who, what, and where types of question (e.g. Alice = PERSON, garden = LOCATION). Heilman and Smith made use of syntactic tools as their main tools for the QG system. However, they also used a semantic-related tool called the Supersense Tagger to generate higher level semantic tags.

There are 3 steps involved in the QG system of Heilman and Smith [18], as displayed in Figure 4.1. The first step, Transformations of Declarative Input Sentences, includes the process of simplifying factual statements and pronoun resolutions. They generated simplified sentences from a Wikipedia article as the input by removing discourse cues.

1. Transformations of Declarative Input Sentences
2. Question Creation
3. Question Ranking

Fig. 4.1.: Steps in the QG System of Heilman and Smith [16] summarized in [18]

Figure 4.2 shows an example of a simplified sentence taken from [18]. In Figure 4.2, the sentence is simplified by removing the discourse marker "however" and the relative clause "which restricted trade with Europe."

Original Sentence:

However, Jefferson did not believe the Embargo Act, which restricted trade with Europe, would hurt the American economy.

Simplified Sentence:

Jefferson did not believe the Embargo Act would hurt the American economy.

Fig. 4.2.: Example of a simplified sentence

The second step in the QG System of Heilman and Smith is Question Creation. The summary of the question creation phase is shown in Figure 4.3.

1. Marking unmovable phrases
2. Generating possible question phrases
3. Decomposition of the main verb
4. Subject-auxiliary inversion
5. Removing answers and inserting question phrases
6. Post processing

Fig. 4.3.: The question creation phase of Heilman and Smith [16] summarized in [18]

In the marking unmovable phrases step, Heilman and Smith created 18 rules in Tregex expressions to avoid the system generates confusing questions. An example is the rule PP

<< PP=unmv to mark prepositional phrases that are nested within other prepositional phrases. Thus, from a sentence "Alice saw the rabbit in the room of hats," the question "What did Alice see the rabbit in the room of?" can be avoided because "the room of hats" cannot be separated. Another example of the rule is NP $ VP << PP=unmv to mark prepositional phrases in subjects. Thus, from a sentence "The capital of Germany is Berlin," the question "What is the capital of Berlin?" can be avoided and instead, the question "What is the capital of Germany?" can be created.

In generating the possible question phrase step, 6 conditions were used to create WH questions (e.g. to create "Where" question, the object of the must be tagged as noun.location with any of the preposition: on, in, at, over, to). The next step, decomposition of the main verb, has several purposes, such as to identify the main clause for subject-auxiliary inversion, and to identify the main verb so that the system can decompose a do or a does form followed by the base form of the verb. The fourth step, subject-auxiliary inversion, is done to generate yes-no questions (e.g. Does Alice like the rabbit?) or when the answer phrase is a non-subject noun phrase (e.g. Who likes the rabbit?) from the sentence "Alice likes the rabbit." In the fifth step, a selected answer phrase is removed and each possible question phrase is inserted into a separate tree. Finally, a post processing step is done to ensure proper formatting such as changing sentences' final periods with question marks, and removing extra white space).

Finally, they included question ranking as the last step in the QG system. They used statistical ranking to the candidates and generate questions with higher ranks. The ranking was done by learning a training set which were prepared by 15 native English-speaking university students.

Figure 4.4 shows the overall process by using a sentence from a Wikipedia article about the history of Los Angeles [18].

4.2.2 Mazidi and Nielsen

The QG system that was developed by Mazidi and Nielsen [24] represents the semantic approach. Their QG system generates the questions by manipulating the predicate and argument structure from semantic role label (SRL). Mazidi and Nielsen used SENNA which

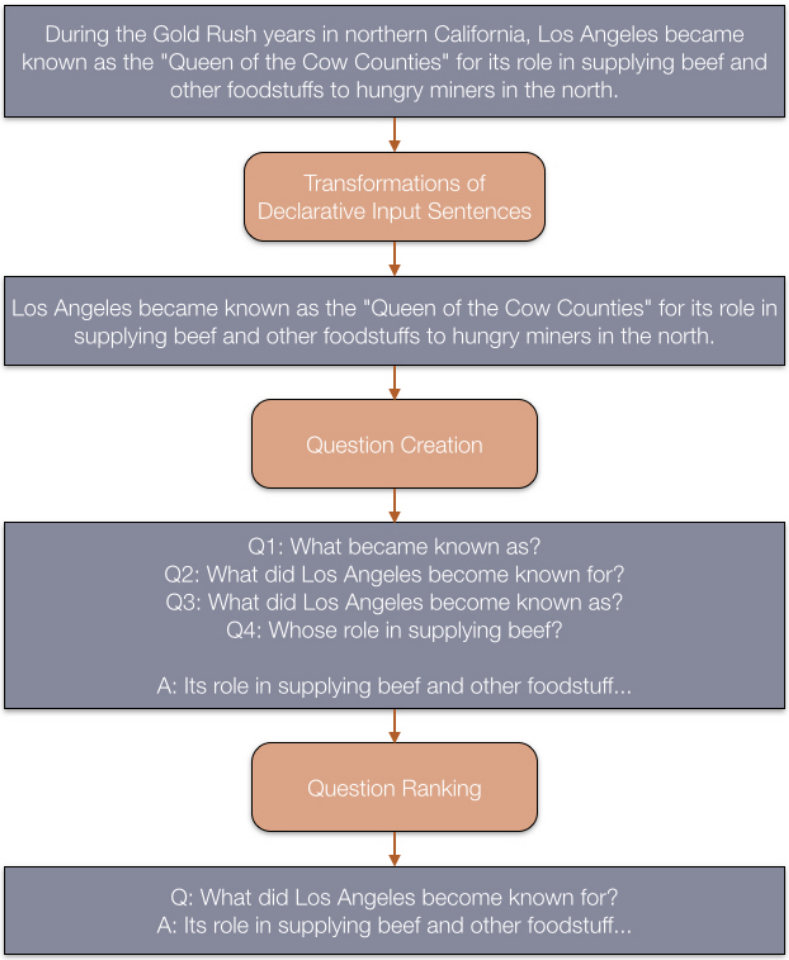

Fig. 4.4.: An example of a generated question and answer pair from the QG system of Heilman and Smith.

simplifies a sentence into several clauses and produces the SRL that identify patterns in the source text.

Besides providing the SRL, SENNA is able to provide POS tagging, chunking, Named Entity Recognition (NER), and syntactic parsing. Figure 4.5 shows the result of SENNA by using a sentence taken from Alice's Adventures in Wonderland chapter 9: "Alice watched the White Rabbit as he fumbled over the list."

The first column shown in figure 19 represents each word in the input, while the second column consists of the Penn Treebank POS tagset [25] of each word:

NNP: Proper noun, singular.

Word	POS Tag	Chunk Tag	NER	Syntactic Parsing	Semantic Role Labelling		
Alice	NNP	S-NP	S-PER	(S1(S(NP*)	-	S-A0	O
watched	VBD	S-VP	O	(VP*	watched	S-V	O
the	DT	B-NP	O	(NP*	-	B-A1	O
White	NNP	I-NP	B-MISC	*	-	I-A1	O
Rabbit	NNP	E-NP	E-MISC	*)	-	E-A1	O
as	IN	S-SBAR	O	(SBAR*	-	B-AM-TMP	O
he	PRP	S-NP	O	(S(NP*)	-	I-AM-TMP	S-A0
fumbled	VBD	S-VP	O	(VP*	fumbled	I-AM-TMP	S-V
over	IN	S-PP	O	(PP*	-	I-AM-TMP	B-AM-TMP
the	DT	B-NP	O	(NP*	-	I-AM-TMP	I-AM-TMP
list	NN	E-NP	O	*))))))	-	E-AM-TMP	E-AM-TMP
.	.	O	O	*))	-	O	O

Fig. 4.5.: The result of POS tagging, chunking, NER, SRL, and syntactic parsing from SENNA

VBD: Verb, past tense.

DT: Determiner.

IN: Preposition or subordinating conjunction.

PRP: Personal pronoun.

NN: Noun, singular or mass.

The third column consists of the chunk tag based on Penn Treebank syntactic tagset [25] with four different prefixes which mark the word position in the segment:

NP: Noun Phrase.

VP: Verb Phrase.

SBAR: Clause introduced by a (possibly empty) subordinating conjunction.

B: beginning.

I: intermediate.

E: ending.

S: a phrase containing a single word.

O: not a member of a chunk.

The fourth column consists of the NER tags - persons, locations, organizations and names of miscellaneous entities - which is assigned on each recognizable named entity. The NER tags also use similar prefixes with the chunk tags to mark the position of the word in the NER phrase. The fifth column consists of the representation of the treebank annotation of the word in the tree. The sixth, seventh, and eighth columns represent sequentially the verb (predicate) of the sentence, and then the predicate-argument structures for each sentence that can be found in the input. The SRL also use similar prefixes with the chunk tags and the NER tags. The predicates in the sentence are labeled as V and the arguments are labeled as A with numbers according to PropBank Frames scheme [26]:
V: verb
A0: agents/causers
A1: patient (the argument which is affected by the action)
AM-TMP: temporal markers

For the question generation process, Mazidi and Nielsen [24] prepared 42 patterns which were based on the PropBank Frames scheme [26]. An example of a pattern that is taken from [26] is shown in Figure 4.6.

Rel: like
Arg0: you
Arg1: [⋆T⋆] -> What

Fig. 4.6.: A Propbank annotation for a WH-phrase

Figure 4.6 shows a pattern that is represented by a Propbank structure for a WH-phrase "What do you like?". In an active phrase "You like cakes", "like" represents the predicate (Rel), while "you" represents the Arg0 and "cakes" represents the Arg1. In the example of WH-phrase shown in Figure 4.6, "like" still represents the Rel and "you" still represents the Arg0. However, the Arg1 is left as a trace.

In the work of Mazidi and Nielsen [24], they prepared a matcher function to match the source sentence's predicate-argument structure - that was previously produced by SENNA - with the list of prepared patterns. Then, they generate questions based on these matched patterns by restructuring the patterns.

In 2015, Mazidi and Nielsen updated their question generation system by combining multiple views of different parsers [23]. The updates involved dependency parsing, SRL, and discourse cues. In order to give a better sense of dependency parsing, an example of a dependency parsing tree is shown in Figure [27].

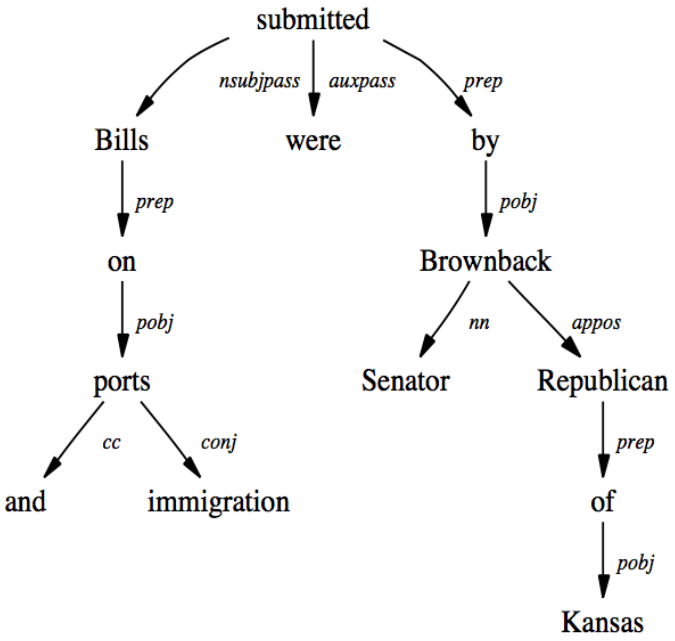

Fig. 4.7.: A dependency parsing tree from the sentence "Bills on ports and immigration were submitted by Senator Brownback, Republican of Kansas" taken from [27].

In their updated system, Mazidi and Nielsen [23] generate the dependency of the source text using the Stanford Parser [27]. They also generate the SRL using SENNA. The results from both the dependency parser and the SRL are then combined.

Figure 4.8 shows the dependency parsing result from the sentence "Alice watched the White Rabbit as he fumbled over the list". By marking the verb "watched" as the root of the tree, the dependency parsing helps to mark the main verb of the sentence, in addition to the semantic role labeling result. In this new system, Mazidi and Nielsen [23] managed to

Fig. 4.8.: The dependency parsing result of "Alice watched the White Rabbit as he fumbled over the list." using Stanford Parser

outperform their previous question generation system by involving the dependency parsing with 21% more semantically-oriented questions versus factoid questions.

4.3 Discussion

Although the initial research on QG focused on the educational or teaching area, recent research has proved that QG can be used for other domains, including the conversational character or virtual human. It can save a lot of time to fill in the domain knowledge for the virtual human rather than manually creating question and answer pairs. It is also good for ARIA-VALUSPA project especially because there are more than one virtual humans that can be developed based on the ARIA-VALUSPA framework. Therefore, a faster and automated process in filling in the domain knowledge is desirable.

However, as pointed out by Yao et al. [1], it should be noted that people can ask different kinds of questions to the virtual human. They might ask a question about something that is not explained in the story; e.g. asking about the appearance of the virtual human, asking about the life of the storys writer. However, QG only creates question and answer pairs from

the information that is provided in the source text. Therefore questions about something that is not in the source text, even if it is still related to the story of Alice in Wonderland, might not be covered using this approach.

Another thing that needs to be considered when using QG is that the generated questions can be too specific. For example: “she soon made out that it was only a mouse that had slipped in like herself”. A possible generated question from this sentence could be “What did Alice find that slipped in like herself?”. For a user to ask this question, he must have a knowledge that Alice is trapped somewhere with someone else.

Lastly, the related works on QG system have implemented different approaches. For example, Heilman and Smith [15] [16] [17] used the syntactic approach while Mazidi and Nielsen [19] used the semantic approach. However, combining information from multiple views can improve the quality of the generated questions as shown by Mazidi and Nielsen [23] by using dependency parsing. Questions that suggest deeper understanding of the main information is more desirable than factual based questions.

5. ALICE QUESTION GENERATION

Alice Question Generation (AQG) is a question generation (QG) system that is developed to generate question and answer pairs about Alice in Wonderland. The generated QA pairs are intended to be stored in the QAMatcher tool (see section 3.2) that can match the stored questions with the questions from the users when they talk with Alice the virtual human. AQG carries the semantic views of text as the main approach for developing the algorithm. However, it also applies the syntactic views to improve the quality of the generated QA pairs. Combining multiple views of text is proven to reduce the error rate of the generated questions [23].

AQG uses semantic role label (SRL) as the main tool to retrieve the semantic meaning of Alice in Wonderland story. SRL is used as the semantic tool because it provides enough information for a sentence to be altered into questions by parsing a sentence into a predicate-argument structure [26]. SENNA is used to retrieve the SRL because the tool can be used easily and it assigns the labels quickly for a number of sentences.

Besides SRL, Stanford Dependency is also used to retrieve the semantic meaning of Alice in Wonderland story. Stanford Dependency is used because it keeps a sentence as a whole without dividing it into clauses, which helps to keep the complete information in a sentence. PyStanfordDependencies is the Stanford Dependency tool that is used for the AQG system. PyStanfordDependencies is used because the library is written in Python, which is the same language as the AQG system, and it is simple enough to be processed by the AQG system.

Figure 5.1 shows an overview of the AQG system. First, SENNA takes an "input" text file consists of the input sentences and produces the SRL in a text file called "output". This process is conducted separately with the AQG system. Next, the AQG system can be run. AQG takes the "input" text file (which is also used by SENNA) and processes them using the PyStanfordDependency library to generate the Stanford dependencies. The result of the dependency is written in an XML file called "Semantic Representation". After this

process, AQG takes the SENNA "output" file and adds the "Semantic Representation" file with the SRL result.

Next, AQG runs the "Template Matching" function which matches the "Semantic Representation" with a number of QA templates. The QA templates are created based on the observation of SRL, which is the main tool that is used as the foundation of AQG. A QA pair is produced every time there is a matching template and is stored in an XML file called "Generated QA". The process of observing the patterns and creating the templates are explained in more detail in the rest of this chapter.

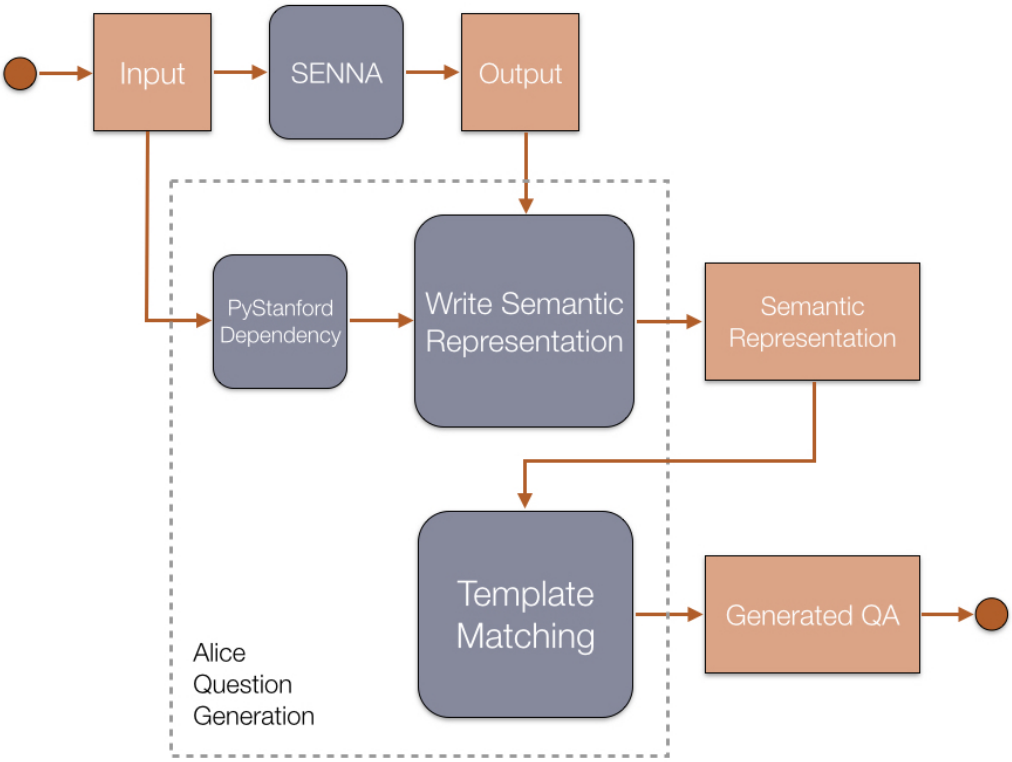

Fig. 5.1.: Overview of the AQG System

5.1 Pattern Observation

The QA templates in AQG are created based on two pattern considerations [28]: the frequency of the pattern occurrences and the consistency of the semantic information conveyed by the pattern across different instances.

Since SRL is used as the main tool to retrieve the semantic meaning of the input, the pattern observation is based on the SRL result. SRL parses a sentence into a predicate-argument structure with consistent argument labels. For example, "the rabbit" is labeled as Arg1 both in "Alice calls the rabbit" and in "The rabbit is called". It also gives labels to all modifiers of the verb, such as temporal (TMP) and locative (LOC).

SENNA [29] is used to determine the SRL of the text input. SENNA divides a sentence into one or more clauses. For example, SENNA divides the sentence "While she is tiny, she slips and falls into a pool of water." into two clauses (see Figure 5.2). The pattern of the first clause "While she is tiny, she slips into a pool of water" is TMP-A1-V-A3, and the pattern of the second clause "While she is tiny, she falls into a pool of water" is TMP-A1-V-A4.

1	While	-	B-AM-TMP	B-AM-TMP
2	she	-	I-AM-TMP	I-AM-TMP
3	is	-	I-AM-TMP	I-AM-TMP
4	tiny	-	E-AM-TMP	E-AM-TMP
5	,	-	0	0
6	she	-	S-A1	S-A1
7	slips	slips	S-V	0
8	and	-	0	0
9	falls	falls	0	S-V
10	into	-	B-A3	B-A4
11	a	-	I-A3	I-A4
12	pool	-	I-A3	I-A4
13	of	-	I-A3	I-A4
14	water	-	E-A3	E-A4
15				

Fig. 5.2.: SRL Representations for "While she is tiny, she slips and falls into a pool of water."

The pattern observation is conducted for all the clauses that are produced by SENNA. The observation is conducted manually. Two summaries of Alice in Wonderland are used

as the training data. The first summary is from GradeSaver[1] and it has 47 sentences, while the second summary is from SparkNotes[2] and it has 56 sentences.

A pattern in a clause always has a verb (V) and at least an argument. The argument can either be a basic argument (Arg, e.g. A0, A1, A2) or a modifier argument (ArgM, e.g. TMP, LOC). Almost all of the clauses in the training data have a V and an Arg; there is only one clause that has a V and an ArgM, without an Arg. Therefore, the algorithm does not include a pattern that has no Arg because it is not frequent. The number of Arg can be one (e.g. only an A0), two (e.g. an A0 and an A1), or even more. In summary, Table 5.1 shows the number of clauses within three conditions of the Arg (Arg>=2, Arg==1, Arg==0).

Table 5.1.: The number of clauses within three conditions of the basic arguments

No	**Pattern**	**Number of Clauses**	**Example of Clause**
1	Arg>=2 ArgM>=0 V==1	222	- Alice (A1) sitting (V) with her sister outdoors (A2) when she spies a White Rabbit with a pocket watch (TMP). - Alice (A0) gets (V) herself (A1) down to normal proportions (A2)
2	Arg==1 ArgM>=0 V==1	64	- She (A0) cried (V) while a giant (TMP). - In the wood (LOC) again (TMP) she (A1) comes (V) across a Caterpillar sitting on a mushroom (LOC)
3	Arg==0 ArgM>=1 V==1	1	- get (V) through the door or too small (DIR) to reach the key (PNC)

[1]Borey, Eddie. "Alice in Wonderland Summary". GradeSaver, 2 January 2001 Web. (accessed April, 24 2017).

[2]SparkNotes Editors. "SparkNote on Alices Adventures in Wonderland." SparkNotes LLC. 2005. http://www.sparknotes.com/lit/alice/ (accessed April 24, 2017).

The first pattern (Arg>=2, ArgM>0, V==1) is included in the algorithm because it is the most frequent pattern in the two summaries. The clauses behind this pattern communicate clear information consistently across the sentences in both summaries. Besides these reasons, two or more Args can make better questions than just one Arg. For example, there are three clauses created from the sentence "Suddenly, the cards all rise up and attack her, at which point she wakes up." Figure 5.3 shows that this sentence creates three clauses with different information:

First clause : Suddenly (ADV) the cards all (A1) rise up (V)

Second clause : the cards all (A0) attack (V) her (A1) at which point she wakes up (TMP)

Third clause : she (A0) wakes (V) up (A2)

1		Suddenly	-	S-AM-ADV	O	O
2		,	-	O	O	O
3		the	-	B-A1	B-A0	O
4		cards	-	I-A1	I-A0	O
5		all	-	E-A1	E-A0	O
6		rise	rise	B-V	O	O
7		up	-	E-V	O	O
8		and	-	O	O	O
9		attack	attack	O	S-V	O
10		her	-	O	S-A1	O
11		,	-	O	O	O
12		at	-	O	B-AM-TMP	O
13		which	-	O	I-AM-TMP	O
14		point	-	O	I-AM-TMP	O
15		she	-	O	I-AM-TMP	S-A0
16		wakes	wakes	O	I-AM-TMP	S-V
17		up	-	O	E-AM-TMP	S-A2
18						

Fig. 5.3.: SRL Representations for "Suddenly, the cards all rise up and attack her, at which point she wakes up."

Even though all three clauses give information, the second clause gives more information than the two other clauses because it has more Args in it, compared to the first and the third clause which only has one Arg. Therefore, the first pattern "Arg>=2, ArgMs>=0, V==1" is chosen to be included in the algorithm.

Besides the basic argument observation, the ArgM is also observed. A pattern in a clause can have or not have an ArgM. There are 8 different ArgMs that occur in both summaries. Table 5.2 shows the ArgMs that occur in the summaries as well as the number of occurrences. The four most frequent ArgMs are used in the templates. They are TMP,

LOC, ADV, and MNR. In conclusion, the patterns that are included in the template creation step is "Arg>=2, ArgM>=0, V==1", and the ArgMs are TMP, LOC, ADV, and MNR. This means that a QA pair can be created when there are 2 or more Args, 0 or more ArgMs (TMP/LOC/ADV/MNR), and a V.

Table 5.2.: Occurrences of the Argument Modifiers

ArgM	**GradeSaver**	**SparkNotes**
TMP (Temporal Markers)	22	16
LOC (Locatives)	8	12
ADV (Adverbials)	9	8
MNR (Manner Markers)	7	17
DIR (Directionals)	6	7
PNC (Purpose, not cause)	2	6
DIS (Discourse Markers)	2	-
MOD (Modals)	1	5

5.2 Template Creation

Based on the pattern observation step, the required elements that can create a QA pair are 2 or more Args, 0 or more ArgMs (TMP/LOC/ADV/MNR), and a V. To make a better QA pair, 4 categories are prepared to group the clauses that have ArgMs. The categories are based on the ArgM because one ArgM can really differ from the other ArgMs. For example, a clause with an ArgM TMP may expect a question word "When", while a clause with an ArgM LOC may need a question word "Where". There is also 1 category created to group the clauses that do not have any ArgMs.

Two or more Args can have different labels. Based on a more detailed observation on the 87 clauses of the first pattern condition, there are 70 patterns that have an A0 and an A1 in its clause. In the PropBank Frames scheme [26], A0 is understood as agents or causers and the A1 is understood as the patient or the one being affected by the action. Therefore, in

the template, the subject character is represented as the lower argument, while the object character is represented as the higher argument.

As a narrative, Alice in Wonderland has the elements that are described in the Elements of a Narrative Theory [30]. Events (actions, happenings) and existents (characters, settings) are the main elements that included in the question generation algorithm. In the implementation, the templates ask about the action that a subject does, the subject character, the object character, and the argument modifier. Based on these narrative elements, there are 5 QA templates that are created for each category that has an ArgM and 4 QA templates that are created for the category without an ArgM. The categories are called MADV, MMNR, MLOC, MTMP, and ARGU. The template names are started with the category name and added with a number.

Table 5.3.: Templates for the category that has an ArgM LOC

Template	**Template Structure**	**Generated QA Pair**
MLOC1	Q: What + aux + lower Arg + do to + higher Arg + ArgM LOC + ? A: lower Arg + V + higher Arg	Q: What does she do to herself in a long hallway full of doors? A: She finds herself
MLOC2	Q: Who + V + higher Arg + ArgM LOC + ? A: lower Arg	Q: Who finds herself in a long hallway full of doors? A: She
MLOC3	Q: What + aux + lower Arg + V + ArgM LOC + ? A: lower Arg + V + higher Arg	Q: What does she finds in a long hallway full of doors? A: She finds herself
MLOC4	Q: What happens to + lower Arg + ArgM LOC + ? A: lower Arg + V + higher Arg	Q: What happens to she in a long hallway full of doors? A: She finds herself
MLOC5	Q: Where + aux + lower Arg + V + higher Arg + ? A: ArgM LOC	Q: Where does she finds herself ? A: in a long hallway full of doors

Table 5.3 shows the 5 QA templates that have been created for MLOC category. The generated QA pairs use the input sentence “She falls for a long time, and finds herself in a long hallway full of doors”. This sentence is divided into two clauses by SENNA:

- She (A1) falls (V) for a long time (TMP)
- She (A0) finds (V) herself (A1) in a long hallway full of doors (LOC)

All of the templates shown in Table 5.3 are created based on the following intentions:
MLOC1: asks about the predicate
MLOC2: asks about the subject
MLOC3: asks about the object
MLOC4: asks about the predicate and the object
MLOC5: asks about the modifier location

The question phrase “What ... do to ...” shown in Table 5.3 is formed for the MLOC1 template because the template asks about the predicate. The lower Arg is located before the phrase “do to” (as the subject) because a lower argument is an agent or a causer. The higher Arg is located after the phrase “do to” (as the object) because a higher argument is the patient or the argument which is affected by the action [26]. The question word “Who” is chosen for the template MLOC2 because most of the subjects in the training data is a character. Moreover, the QAMatcher usually still matches a question correctly even though it uses a different question word. Figure 5.4 shows this example.

```
Question:who form a pool on the floor of the hall?
Best answer :her big tears  does
Question:what form a pool on the floor of the hall?
Best answer :her big tears  does
```

Fig. 5.4.: Two different question words are given a same answer

The generated QA pairs that are shown in Table 5.3 have several syntax errors. They are shown in the template MLOC3 and MLOC5. The verb “find” should be generated instead of “finds”. However, syntax errors or small grammar errors are not handled by the AQG system because the QAMatcher can still match a question correctly when there is a

small syntax error. There is another error that is shown in the generated QA pair from the template MLOC4. The object in the question "what happens to she" is supposed to use an objective pronoun "her" instead of a subjective pronoun "she". The handle for the subjective and objective pronoun is implemented in the next version of the templates.

SENNA usually divides a sentence into several clauses. This makes some information in a sentence missing, especially in a sentence with conjunctions. Using the previous example, there will not be a QA pair that gives an information about how she finds herself in a long hallway full of doors all of a sudden, despite the fact that the reason "she falls for a long time" is given in the same sentence. This creates a new situation in which a dependency parse can be useful. Therefore, a new template under a new category is created. The template uses dependency parsing and requires a conjunction in the sentence. A Python interface called PyStanfordDependencies is used to provide the Stanford Dependencies in AQG. Based on the observations of dependency labels on sentences with conjunctions, the new template is as follows:

Question = 'What happens when ' + Subj + V + Dobj + Nmod + '?'

Answer = Subj + V + Dobj + Nmod + Cc + Conjs

For example, the sentence "She falls for a long time, and finds herself in a long hallway full of doors" has the dependency result which is shown in Figure 5.5.

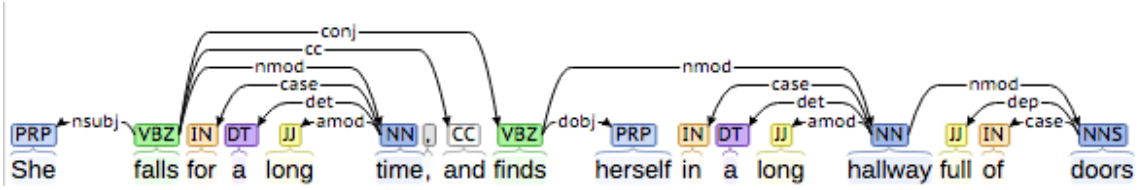

Fig. 5.5.: Dependency Parse Result for the Sentence "She falls for a long time, and finds herself in a long hallway full of doors"

Therefore, a new question and answer pair that is generated by the algorithm is:

Q: What happens when she falls for a long time?

A: She falls for a long time and finds herself in a long hallway full of doors

In summary, all categories that are created are displayed in Table 5.4 with their required elements and the number of templates. In total, there are 25 templates that fall into 6

categories. The structures of initial templates for all categories are displayed in Table A.1 in the Appendix.

Table 5.4.: Categories and Templates

Category Name	**Required Elements**	**Total Templates**
MADV	Arg>=2, ArgM==ADV, V==1	5
MMNR	Arg>=2, ArgM==MNR, V==1	5
MLOC	Arg>=2, ArgM==LOC, V==1	5
MTMP	Arg>=2, ArgM==TMP, V==1	5
ARGU	Arg>=2, ArgM==0, V==1	4
DCNJ	Conj>=1	1

6. INITIAL EVALUATION AND IMPROVEMENT

First of all, a simple "QA Grouping" algorithm is created to group all the generated QA pairs based on their categories and to store each category in a CSV file. There are 6 CSV files generated based on the training data and can be viewed and analyzed easily using spreadsheet applications. Next, a pre-initial evaluation is conducted to see if the program works and if all the templates do not create too much error. The pre-initial evaluation is explained in section 6.1. Next, an initial evaluation is conducted to measure the quality of the initial templates. The initial evaluation is explained in section 6.2. The pre-initial evaluation and the initial evaluation are conducted by the author. Finally, an error analysis and improvements are next conducted based on the result of the initial evaluation. The error analysis and improvements are explained in section 6.3.

6.1 Pre-Initial Evaluation

A quick pre-initial evaluation is conducted by using one summary from the training data, the GradeSaver summary. There are 435 QA pairs that are generated from 47 sentences of the summary. Based on the observation of the generated question and answer of this initial version, there are 6 templates that create too many strange results.

Table 6.1 shows the templates that create too many errors. It seems too difficult to create a good template that asks about the elements that these templates were meant to ask. For the MMNR category, the verb is related to the MNR because MNR modifies the verb, instead of the entire sentence like an ADV modifier [26]. When altering the pattern to create the template, it is important to keep the verb with the ArgM MNR, and thus make limitations on the templates that can be created. For example, the question that is generated from the template that asks about the verb and the object, MMNR4: "What happens to she through this door?". The phrase "through this door" explains how she does the "spy" activity. Since the ArgM Manner "through this door" is separated from the verb, it makes the question sound strange. The template that asks about the verb and the object

Table 6.1.: Templates that Creates Too Many Errors

Template	**Description**	**Examples**
MMNR1	Asks about the action/ verb	Q: What does she do to herself down enough ? A: she shrinks herself
MMNR3	Asks about the object	Q: What does she finds with a note that asks her to drink it ? A: she finds a drink
MMNR4	Asks about both the action /verb and the object	Q: What happens to she through this door ? A: she spies a beautiful garden
ARGU1	Asks about the action/ verb	Q: What does He do to her? A: He mistakes her Q: What does He do to her? A: He sends her
ARGU3	Asks about the object	Q: What does she get? A: she get a handle Q: What does she get? A: she get herself
ARGU4	Asks about both the action /verb and the object	Q: What happens to Alice? A: Alice grow larger and smaller Q: What happens to Alice? A: Alice takes the baby

from the MADV category, however, generates a better structured question. For example, the question "What happens to she while in the white rabbit's home?" and the answer "she becomes too huge to get out through the door" are generated from the template MADV4.

There is another error that can be seen from the generated QA pairs from the Table 6.1, which is the objective pronoun. The objective pronoun error "what happens to **she**" instead of "what happens to **her**" is fixed in the next version of the template.

For the ARGU category, the error that can be found is that the category only provides two Args and one V and makes the generated questions too vague. For example, the question "what does she get" is generated 5 times with different answers according to different scenarios in the story. Since there is no ArgM in ARGU category, the case for the generated QA is not specific enough. In conclusion, these 6 templates are removed from AQG.

6.2 Initial Evaluation

The evaluation that is conducted for the AQG system uses a rating scheme which is developed to be easy for novice annotators [18]. This is because the users who will interact with the virtual human can be general people without advanced knowledge in linguistic. For this evaluation, each question and answer pair is rated by the author on a 1 to 5 scale as displayed in Table 6.2.

Table 6.2.: 5 Scale Acceptability Score Adapted from [18]

Scale	**Score Explanation**
Good (5)	The QA pair does not have any problems, and it is a good as the one that a person might ask and the virtual human might answer.
Acceptable (4)	The QA does not have any problems..
Borderline (3)	The QA might have a problem, but I'm not sure.
Unacceptable (2)	The QA definitely has a minor problem.
Bad (1)	The QA has major problems.

There are 19 templates that are further evaluated. Two summaries of the training data are used for the initial evaluation. The first one is a summary from GradeSaver which consists of 47 sentences, and the second one is a summary from SparkNotes which consists

of 56 sentences. The score for the question and answer pair acceptability is displayed in Figure 6.1.

Category	Number of templates	GradeSaver		SparkNotes	
		Q&A pairs	Average score	Q&A pairs	Average score
MADV	5	30	3.100	30	2.767
MMNR	2	10	3.400	10	3.500
MLOC	5	15	3.267	20	3.100
MTMP	5	65	2.246	40	2.675
ARGU	1	70	2.729	121	2.835
DCNJ	1	20	3.650	31	3.129
Total	19	210	**2.790**	252	**2.885**

Fig. 6.1.: Initial Evaluation Result

As shown in Figure 6.1, the overall question and answer pairs are still below the borderline scale (3), which are 2.790 and 2.885. Next, an error analysis is conducted and continued by template improvements.

6.3 Error Analysis and Template Improvement

After conducting the initial evaluation, the errors from each category are analyzed. The templates are then improved based on the result of the error analysis. The list of the improved templates are displayed in Table A.2 in the Appendix. The error analysis and the template improvements are explained in the rest of this section.

6.3.1 MADV

The average score of MADV category for the GradeSaver summary, 3.1, is slightly better than the average score for the SparkNotes summary which is 2.767. However, when observing the lower scores in the result of both summaries, there are several things that can be improved on the template. The analysis can be explained by the examples in Figure 6.2.

No	Template	Clause	Question	Answer
1	MADV1	Fascinated by the sight (ADV), she (A0) follows (V) the rabbit (A1) down the hole (A2)	What does she do to the rabbit Fascinated by the sight ?	she follows the rabbit
2	MADV1	Left alone (ADV), she (A1) goes (V) on (DIR) through the wood (A2) and runs into the White Rabbit	What does she do to through the wood Left alone ?	she goes through the wood
3	MADV1	As she cries (ADV), Alice (A1) shrinks and falls (V) into the pool of tears (A4)	What does Alice do to into the pool of tears As she cries ?	Alice falls into the pool of tears
4	MADV2	Left alone (ADV), she (A1) goes (V) on (DIR) through the wood (A2) and runs into the White Rabbit	Who goes through the wood Left alone ?	she
5	MADV3	Fascinated by the sight (ADV), she (A0) follows (V) the rabbit (A1) down the hole (A2)	What does she follows Fascinated by the sight ?	she follows the rabbit
6	MADV4	Fascinated by the sight (ADV), she (A0) follows (V) the rabbit (A1) down the hole (A2)	What happens to she Fascinated by the sight ?	she follows the rabbit
7	MADV5	Fascinated by the sight (ADV), she (A0) follows (V) the rabbit (A1) down the hole (A2)	When does she follows the rabbit ?	Fascinated by the sight

Fig. 6.2.: The Initial Evaluation Result for the MADV Category

Figure 6.2 shows the template name, the clauses which were used as the input of the AQG system, and the generated questions and answers of the clauses. The first clause explains about how Alice follows a white rabbit when she was fascinated by the sight. Therefore, a word "when" would be better added before "fascinated by the sight" to make the question clearer. The same solution can also apply for the second clause. For the second clause, the question "What does she do through the wood **when** she was left alone?" would sound better.

Another problem that remains, however, is that the verb "do to" can only fit clauses with Arg number A0 and A1. A0 and A1 both are the dominant Args that consist in the training data (the GradeSaver and the SparkNotes summary) as mentioned in section 5.2, but generalizing the template by using "do to" can causes errors in several clauses. Leaving the object from the question template can be the solution for this. Therefore, the question

"What does she do when she was left alone" would be a better generated question that can still be used for the first example as well: "What does she do when she was fascinated by the sight".

Using the conjunction "when" before the ArgM apparently can cause a problem for the third clause, despite the fact that it would be good for the first and the second clause. The phrase "as she cries" is labeled as the ArgM ADV. It means that when using "when" the question will become "What does Alice into the pool of tears when as she cries?". In order to handle this problem, the syntax is checked further. If the ArgM ADV starts with "as", then the conjunction "when" is changed into "as" instead. This also applies to other ArgM ADV that start with "while", "to", and "into".

Adding the "when" conjunctions, or having the first word of the ArgM as the conjunctions for the word "as", "while", "to", and "into" applies to questions in the templates MADV1, MADV2, MADV3, and MADV4, and also applies to the answer in the template MADV5. Based on the changes for the QA template under MADV category, the new result from the clauses in Figure 6.2 is displayed in Figure 6.3.

6.3.2 MMNR

The average score for the MMNR category is higher compared to the MADV category. This is because on the pre-initial evaluation, the templates MMNR1, MMNR3, MMNR4 were removed. Therefore, only the generated QA pairs from MMNR2 and MMNR5 category still remain and their quality is pretty good.

Despite their high score when compared to the other categories, there are two improvements done for the MMNR category. They can be explained using the generated QA pairs shown in Figure 6.4.

The second clause in 6.4 shows that she finds a drink that has a note on the drink. The phrase "with a note that asks her to drink it" actually refers to "a drink" instead of the verb "drink". The phrase "with a note that asks her to drink it" is not supposed to be labeled as the manner modifier of the verb "finds", because manners adverbs specify how an action is performed [26]. This also happens on the third clause. The phrase "with a door" refers to the "tree" instead of the verb "finds". This makes the question template

No	Template	Clause	Question	Answer
1	MADV1	Fascinated by the sight (ADV), she (A0) follows (V) the rabbit (A1) down the hole (A2)	What does she do when Fascinated by the sight ?	she follows the rabbit down the hole
2	MADV1	Left alone (ADV), she (A1) goes (V) on (DIR) through the wood (A2) and runs into the White Rabbit	What does she do when Left alone ?	she goes through the wood
3	MADV1	As she cries (ADV), Alice (A1) shrinks and falls (V) into the pool of tears (A4)	What does Alice do As she cries ?	Alice falls into the pool of tears
4	MADV2	Left alone (ADV), she (A1) goes (V) on (DIR) through the wood (A2) and runs into the White Rabbit	Who goes through the wood when Left alone ?	she goes through the wood
5	MADV3	Fascinated by the sight (ADV), she (A0) follows (V) the rabbit (A1) down the hole (A2)	What is it that she follows when Fascinated by the sight ?	she follows the rabbit down the hole
6	MADV4	Fascinated by the sight (ADV), she (A0) follows (V) the rabbit (A1) down the hole (A2)	What happens to her when Fascinated by the sight ?	she follows the rabbit down the hole
7	MADV5	Fascinated by the sight (ADV), she (A0) follows (V) the rabbit (A1) down the hole (A2)	When does she follows the rabbit down the hole ?	when she is Fascinated by the sight

Fig. 6.3.: The Generated QA Pairs of the MADV Category After the Improvements

No	Template	Clause	Question	Answer
1	MMNR2	Through the door (MNR), she (A0) sees (V) a beautiful garden (A1)	Who sees a beautiful garden Through the door ?	she
2	MMNR5	she (A0) finds (V) a drink (A1) with a note that asks her to drink it (MNR)	How does she finds a drink ?	with a note that asks her to drink it
3	MMNR5	She (A0) finds (V) a tree (A1) with a door (MNR)	How does She finds a tree ?	with a door

Fig. 6.4.: Errors on the Initial Evaluation Result of the MMNR Category

"How + does + lower Arg + V + higher Arg" not fit in the clause with an ArgM MNR that starts with "with". This template, however, still works in other clauses, such as: Clause: Through the door (MNR), she (A0) sees (V) a beautiful garden (A1)

Question: How does she sees a beautiful garden ?
Answer: Through the door

Considering this problem, therefore this template is still kept as it is; however, this template will not generate a QA pair when the ArgM MNR starts with "with".

Another change that is done for the MMNR category is that on the MMNR2 template, the answer is added with an auxiliary, instead of just a "higher Arg". Therefore, the answer for "Who sees a beautiful garden through the door" is "she does."

6.3.3 MLOC

The average score for the MLOC category is above the borderline, 3.267 and 3.1 for the GradeSaver and the SparkNotes summary respectively. However, there are still some improvements conducted for this category which can be explained using 5 generated QA pairs displayed in Figure 6.5.

No	Template	Clause	Question	Answer
1	MLOC1	She (A0) finds (V) herself (A1) in a long hallway full of doors (LOC)	What does She do to herself in a long hallway full of doors ?	She finds herself
2	MLOC1	A key (A1) she (A0) discovers (V) on a nearby table (LOC)	What does she do to a key on a nearby table ?	she discovers a key
3	MLOC2	She (A0) finds (V) herself (A1) in a long hallway full of doors (LOC)	Who finds herself in a long hallway full of doors ?	She
4	MLOC3	The forest (LOC) where (R-LOC) she (A0) meets (V) a Caterpillar sitting on a mushroom and smoking a hookah (i.e., a water pipe) (A1)	What does she meets the forest ?	she meets a Caterpillar sitting on a mushroom and smoking a hookah (i.e. , a water pipe)
5	MLOC4	Her giant tears (A0) form (V) a pool (A1) at her feet (LOC)	What happens to her giant tears at her feet ?	her giant tears form a pool

Fig. 6.5.: Errors on the Initial Evaluation Result of the MLOC Category

The phrase "do to" often implies a subject that is doing a negative action towards the object, such as "what does she do to him?", or "what does he do to the cat?". The

question “What does she do to herself” on the first example in Figure 6.5 therefore can turn into other interpretations instead of the actual fact that she only finds herself, and not doing anything to herself. The second question, “What does she do to a key” also not the best question given the phrase “she discovers a key” as the source sentence. Based on this problem, “do to” and the object are removed from the template, leaving only the subject and the ArgM LOC on the question template.

Next, “what does she meets the forest” lacks a preposition before the ArgM LOC. This is because “the forest” is referred by the relative adverb “where” which is left out by AQG. To handle this problem, a preposition “in” is added to every ArgM LOC that does not have any prepositions.

Another problem can be found on the fifth question. The question “what happens to her giant tears at her feet” given the answer “her giant tears form a pool” can be made less strange. On this MLOC4 template, the subject is left out for the improved version of the template. Therefore, the generated question from the clause “Her giant tears form a pool at her feet” becomes “What happens at her feet?”. After improving the MLOC4 template, other clauses can still have good results, for example:
Clause: A key (A1) she (A0) discovers (V) on a nearby table (LOC)
Question: What happens on a nearby table ?
Answer: she discovers a key

Finally, a small improvement is done for MLOC2 template. The answer template is extended with an auxiliary. Therefore, the answer “she does” is generated, with a question pair “Who finds herself in a long hallway full of doors”. Figure 6.6 displays the result after the template improvements.

6.3.4 MTMP

The MTMP category has an average score that is lower than the borderline. Some problems that are found in this category can be explained using the clauses that are displayed in Figure 6.7.

No	Template	Clause	Question	Answer
1	MLOC1	She (A0) finds (V) herself (A1) in a long hallway full of doors (LOC)	What does She do in a long hallway full of doors ?	She finds herself
2	MLOC1	A key (A1) she (A0) discovers (V) on a nearby table (LOC)	What does she do on a nearby table ?	she discovers a key
3	MLOC2	She (A0) finds (V) herself (A1) in a long hallway full of doors (LOC)	Who finds herself in a long hallway full of doors ?	She does
4	MLOC3	The forest (LOC) where (R-LOC) she (A0) meets (V) a Caterpillar sitting on a mushroom and smoking a hookah (i.e., a water pipe) (A1)	What does she meets in the forest ?	she meets a Caterpillar sitting on a mushroom and smoking a hookah (i.e. , a water pipe)
5	MLOC4	Her giant tears (A0) form (V) a pool (A1) at her feet (LOC)	What happens at her feet ?	her giant tears form a pool

Fig. 6.6.: The Result After the Improvement on the MLOC Category

No	Template	Clause	Question	Answer
1	MTMP1	She (A0) a little cake (A0) which (R-A0) when eaten (TMP) makes (V) her small (A1) again (TMP)	What does She do a little cake when eaten ?	She makes a little cake
2	MTMP2	She (A0) a little cake (A0) which (R-A0) when eaten (TMP) makes (V) her small (A1) again (TMP)	Who makes a little cake when eaten ?	She
3	MTMP3	The cards all (A0) attack (V) her (A1) at which point she wakes up (TMP)	What does the cards all attack at which point she wakes up ?	the cards all attack her
4	MTMP3	All of a sudden (TMP), Alice (A0) finds (V) herself (A1)	What does Alice finds All of a sudden ?	Alice finds herself

Fig. 6.7.: Errors on the Initial Evaluation Result of the MTMP Category

The clauses on the first and the second result are from the same sentence: “She eventually finds a little cake which, when eaten, makes her small again”. SENNA labels the pronoun “She” and “a little cake” with the same “A0”. It is wrong and it makes the generated QA pair strange that it cannot be understood: “What does she do a little cake when eaten”. Therefore, on the improved templates, AQG leaves out all clauses that have two

phrases with a same number of arguments. This condition is also implemented in all the templates in all categories.

For the template MTMP3, the question word "What" is changed into "Whom" since most of the objects in the clauses in the training data are characters. Therefore, a "who" is more suitable for the objects.

Another improvement is also done for the template MTMP4, in which the subject is left out after the question "What happens". Therefore, the new question template is "What happens + ArgM TMP?". This improvement is similar with the one for MLOC4 question template.

6.3.5 ARGU

The generated QA pairs that are displayed in Figure 6.8 somehow imply incorrect information according to the given clause. For example, the generated question from the second clause, "Who make Alice" sounds like it asks about how Alice existed today. However, the original clause talks about how two mushrooms can make Alice grow larger and smaller.

No	Template	Clause	Question	Answer
1	ARGU2	her (A0) fetch (V) some things (A1) from his house (A2)	Who fetch some things ?	her
2	ARGU2	the two sides (A0) make (V) Alice (A1) grow larger and smaller (A2)	Who make Alice?	the two sides
3	ARGU2	different parts of the mushroom (A0) make (V) her (A1) grow (A2)	Who make her?	different parts of the mushroom

Fig. 6.8.: Errors on the Initial Evaluation Result of the ARGU Category

The obvious improvement that can be done for this is to include the Arg 2 (A2) from the clause into the generated question. Therefore, the question "Who make Alice?" become clearer in "Who make Alice grow larger and smaller?". It is important to keep the A2 in the generated questions. Figure 6.9 shows the result after improving the template.

No	Template	Clause	Question	Answer
1	ARGU2	her (A0) fetch (V) some things (A1) from his house (A2)	Who fetch some things from his house ?	her does
2	ARGU2	the two sides (A0) make (V) Alice (A1) grow larger and smaller (A2)	Who make Alice grow larger and smaller ?	the two sides does
3	ARGU2	different parts of the mushroom (A0) make (V) her (A1) grow (A2)	Who make her grow?	different parts of the mushroom does

Fig. 6.9.: The Result After the Improvement on the ARGU Category

6.3.6 DCNJ

The category DCNJ has high average scores too for the two summaries, compared to the other categories. However, some improvements are still made for this category.

No	Template	Clause	Question	Answer
1	DCNJ1	She longs to get there , but the door is too small	What happens when longs get ?	longs get but the door is too small ;
2	DCNJ1	While stretched out , she pokes her head into the branches of a tree and meets a Pigeon	What happens when she pokes head into the branches tree ?	she pokes head into the branches tree and meets a Pigeon ;
3	DCNJ1	Present are the March Hare , the Hatter , and the Dormouse	What happens when Hare ?	Hare and the Hatter ;the Dormouse ;
4	DCNJ1	This time , she is prepared , and she manages to get to the lovely garden that she saw earlier	What happens when This she prepared ?	This she prepared and she manages to get to the lovely garden that she saw earlier ;
5	DCNJ1	The Knave of Hearts has been accused of stealing the tarts of the Queen of Hearts , but the evidence against him is very bad	What happens when accused ?	accused but the evidence against him is very bad ;

Fig. 6.10.: Errors on the Initial Evaluation Result of the DCNJ Category

The first generated question that is shown in Figure 6.10 is difficult to understand. When looking into the parsing result, “longs” is incorrectly labeled by PyStanfordDependencies. It is incorrectly labeled as another NSUBJ while “get” is labeled as the root (see Figure 6.11). This makes the template assign “longs” as the subject, and thus generate the question

"What happens when longs get?". To generate a better question, "She" should be the NSUBJ while "longs" should be the root, as parsed by another Stanford Dependency Parser that is visualized by the Brat tool [1] illustrated in Figure 6.12.

```
Token(index=1, form='She', cpos='PRP', pos='PRP', head=4, deprel='nsubj')
Token(index=2, form='longs', cpos='NNP', pos='NNP', head=4, deprel='nsubj')
Token(index=3, form='to', cpos='TO', pos='TO', head=4, deprel='mark')
Token(index=4, form='get', cpos='VB', pos='VB', head=0, deprel='root')
Token(index=5, form='there', cpos='RB', pos='RB', head=4, deprel='advmod')
Token(index=6, form=',', cpos=',', pos=',', head=4, deprel='punct')
```

Fig. 6.11.: Incorrect Dependency Parsing Result for the sentence "She longs to get there, but the door is too small"

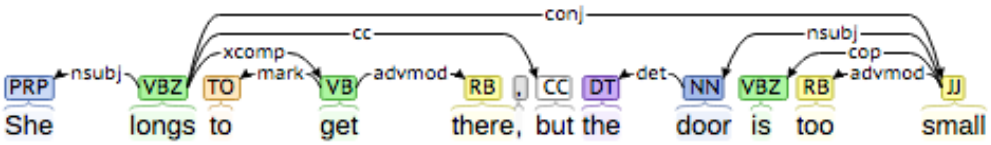

Fig. 6.12.: Correct Dependency Parsing Result for the sentence "She longs to get there, but the door is too small"

Another mistake with parsing the dependency also happens on the third clause "Present are the March Hare, the Hatter, and the Dormouse". In the dependency result, "Hare" is labeled as the root, despite of the fact that "the March Hare" is a character.

The fourth clause in Figure 6.10 also has been incorrectly parsed by PyStanfordDependencies. "This time" is parsed as NSUBJ to the root "prepared", as well as the NSUBJ "she". This makes the AQG pick a strange subject for the generated question "this she" ("this" is a determiner before the NSUBJ "time").

Another improvement is conducted for the problem in the second clause shown in Figure 6.10. The generated question "what happens when she pokes head" lacks the possession modifier, despite its existence in the clause "she pokes her head". This is, however, included in the improved version of the template. After the improvement, the generated QA from the second clause is:

[1] http://nlp.stanford.edu:8080/corenlp/process

Question: What happens when she pokes **her** head into the branches tree?

Answer: she pokes **her** head into the branches tree and meets a Pigeon

Finally, "the Knave of Hearts" is not included in the generated question as shown in Figure 6.10. However, the initial version of the template leaves out passive phrases. The initial template does not check for the passive dependency labels NSUBJPASS, which is the label for "Knave". Therefore, this is included in the improved version of the template. After the improvement, the generated QA from the fifth clause is:

Question: What happens when **the Knave of Hearts** accused?

Answer: **The Knave of Hearts** accused but the evidence against him is very bad

6.4 Evaluation After Template Improvements

After error analysis and template improvement, another evaluation is conducted by the author using the 5-Score Scale explained in Table 6.2. The input source are still the same, they are the GradeSaver and the SparkNotes summary. The number of input sentences are thus the same with the initial evaluation. However, fewer QA pairs are generated than the initial one because of the template improvement. Such as, not generating a QA pair when there is a clause with more than one Args with the same number.

Category	Number of templates	GradeSaver		SparkNotes	
		Q&A pairs	Average score	Q&A pairs	Average score
MADV	5	30	4.367	30	3.567
MMNR	2	9	3.800	9	3.667
MLOC	5	15	3.467	20	4.250
MTMP	5	60	3.467	40	3.125
ARGU	1	60	3.450	99	3.717
DCNJ	1	20	4.250	31	4.097
Total	19	194	**3.696**	229	**3.690**

Fig. 6.13.: Evaluation Result after the Template Improvement

The average score for overall summaries is now increased to 3.696 and 3.690 as displayed in Figure 6.13. This means that the average score is above the borderline score (3) after the improvement.

7. USER EVALUATION OF ALICE QUESTION GENERATION

After the initial evaluation and the improvements, an evaluation with external annotators is conducted. The evaluation measurement is explained in section 7.1. The evaluation setup is explained in section 7.2. Finally, error analysis and improvements for the templates are again conducted based on the result. The error analysis and improvements are explained in section 7.3

7.1 Evaluation Measurement

An evaluation with external annotators are conducted to rate the generated QA pairs from the improved templates. The 5-scale rating system displayed in Table 6.2 is again used for the evaluation. When a QA pair is rated as unacceptable (2) or bad (1), the annotator can choose one or both of the reasons that are shown in Table 7.1.

Table 7.1.: Reasons for an Unacceptable or a Bad Score

Reason	**Reason Explanation**
Incorrect Information (a)	The Q&A implies something that is obviously incorrect according to the context
Awkwardness/Other (b)	The Q&A is awkwardly phrased or has some other problem (e.g., no native speaker of English would say it this way, or the question word is wrong).

The first reason that is displayed in 7.1 can be chosen when the generated question or answer does not entail correct information that is given in the sentence. The following sentence for example: “There is later a cake with a note that tells her to eat; Alice uses both, but she cannot seem to get a handle on things, and is always either too large to get

through the door or too small to reach the key" generates the following QA pair:
Question: Who get a handle on things ?
Answer: she does
An Unacceptable score (2) can be assigned to the above QA pair because the correct information from the sentence is "but she cannot seem to get a handle on things". A question "Who cannot get a handle on things?" or an answer "nobody" should have been generated for this case.

The second reason, awkwardness/other, is more related to the structure of the generated QA. Therefore, it can be assigned when the phrase of the generated QA is strange or when the question word is wrong. For example, the sentence "She longs to get there, but the door is too small" generates the following QA pair:
Question: What happens when She longs get ?
Answer: She longs get but the door is too small

A bad score (1) can be assigned to above QA pair because the structure of the question and answer is strange and they are difficult to understand.

7.2 Evaluation Setup

A summary of Alice in Wonderland from Litcharts[1] is used as the test data. This summary consists of 69 sentences. There are 6 annotators involved in this evaluation. All annotators are students or recently graduated students from English taught programmes at the University of Twente, and one from the Saxion University of Applied Sciences. All annotators are not native speakers, however, they understand English well and speak English almost in daily life.

The test data is separated into two parts. The first half consists of 35 sentences and the second half consists of 34 sentences. The first half of the QA pairs is evaluated by 3 annotators. The second half of the QA pairs is also evaluated by 3 annotators. The annotators are assigned randomly to either the first half group or the second half group. The 35 sentences from the first half group have 137 generated QA pairs, and the 34 sentences

[1]http://www.litcharts.com/lit/alice-s-adventures-in-wonderland/summary

from the second half group have 131 generated QA pairs. The generated QA pairs in each group are also randomized.

No	Sentence	Question	Answer	Score*	Reason**	Comment***
12	She finally finds herself in the beautiful garden she has been aiming for	Where does She finds herself?	in the beautiful garden she has been aiming for	5		
13	He begins to describe the day in question , but keeps getting cut off by the Hare and the King threatens him with execution and calls the next witness	Who getting cut ?	He does	2	a	
14	Alice then starts to tell her story and again finds that she has forgotten certain rhymes and songs , so she gives up telling her adventures and the Mock Turtle starts a song about soup	Who tell her story ?	Alice does	5		
15	She soon comes upon a tree , with a tiny door , and uses the shrinking mushroom to get to the right size to go in	What happens when She comes soon upon a tree with a tiny door ?	She comes soon upon a tree with a tiny door and uses the shrinking mushroom to get to the right size to go in	3		Question repeated in answer
16	The King of Hearts is acting as the judge and the jurors are a collection of dim witted animals	What happens when The King of Hearts acting as the judge ?	The King of Hearts acting as the judge and the jurors are a collection of dim witted animals	1	a	We don't know what happens

Fig. 7.1.: User's Evaluation Form

First, the annotators are given an instruction, an explanation of the 5-scale scoring system, an explanation of both reasons for the unacceptable or bad score, and some examples of rated QA (see Appendix chapter B). In the instruction, the annotators are told to not focus on the grammars and pronoun resolution for the current evaluation. The grammar errors and pronouns are not handled in the AQG because the QAMatcher can still correctly match the questions. However, when they find the generated QA pairs have grammars or pronoun resolutions that are too much for them to understand the generated QA pairs, then they can include them in their judgments.

Next, the annotators give a score for each QA pair. The time that the annotators take to finish the task is about one hour. Figure 7.1 shows 5 examples of generated QA pairs that have been rated by an annotator.

7.3 Error Analysis and Template Improvement

Figure 7.2 displays the average score from all annotators. Based on this user evaluation, error analysis and improvements are again conducted for the templates. Almost all the errors found are improved, except for the ones that cannot be fixed because of parser errors

Category	First Half		Second Half	
	Q&A pairs	Average score	Q&A pairs	Average score
MADV	15	3.111	20	2.583
MMNR	6	3.333	4	2.500
MLOC	10	4.167	20	3.767
MTMP	30	3.267	20	3.667
ARGU	63	3.614	51	3.810
DCNJ	13	3.103	16	3.604
Total	137	**3.462**	131	**3.529**

Fig. 7.2.: User Evaluation Score Result of the Question and Answer Rating

or sentence ambiguities. However, several minor errors on generated questions are not fixed if the QAMatcher can still match the questions well (such as a wrong question word).

7.3.1 MADV

The majority of the problems that exist in the MADV category are caused by strange parsing results. An example of parsing error is displayed in Figure 7.3.

Clause: The White Rabbit returns (A0) having lost his gloves, and, mistaking Alice for his maid (ADV) asks (V) her fetch them (A1)
Question: When does The White Rabbit returns asks her fetch them ?
Answer: when The White Rabbit returns is having lost his gloves , and , mistaking Alice for his maid

Fig. 7.3.: SRL Labeling Error from the MADV Category

Figure 7.3 shows a parsing error on the subject. The verb “returns” is not supposed to be labeled as the subject together with “The White Rabbit”. It makes the generated

question and answer sound strange. Beside of the parsing error, the clause also has a grammar mistake by not including the preposition “to” before “fetch them”.

Another problem that can be seen from the evaluation result, which was also pointed out by an annotator, is a causality problem in one of the sentences that falls under the MADV category. The sentence and the generated QA are displayed in Figure 7.4.

Clause: The Queen (A0) gets (V) very irate (A1) calling for mass executions (ADV)
Question: Who gets very irate when calling for mass executions ?
Answer: The Queen gets very irate

Fig. 7.4.: Causality Problem in a Generated QA from the MADV Category

What the clause in Figure 7.4 trying to say is the queen gets very irate, and then she calls for mass executions. However, this clause can also mean that the queen gets very irate when she calls for mass executions. Unfortunately, the parser gets the latter meaning of the clause which makes the clause fall into the MADC category and generates a question that implies incorrect information.

7.3.2 MMNR

A parsing error also occurs in a clause under the MMNR category as displayed in Figure 7.5. The complete sentence of the clause is “She tries one side of the mushroom and finds it makes her smaller so quickly eats the other side, which makes her grow taller, but mostly in the neck”. The phrase “her smaller” is incorrectly labeled as the subject (Arg 0) which creates a strange answer as shown in Figure 7.5.

Clause: her smaller (A0) so quickly (MNR) eats (V) the other side, which makes her grow taller, but mostly in the neck (A1)
Question: Who eats the other side , which makes her grow taller , but mostly in the neck so quickly ?
Answer: her smaller does

Fig. 7.5.: Parsing Error from the MMNR Category

Besides parsing errors, sentence ambiguity is another source of low scores in two QA pairs which are generated from the same sentence, as shown in Figure 7.6. The generated QA pairs are correct. Playing card is actually a character in the story. The parser labeled the playing cards correctly as the subject, however, it might be confusing for the annotators since the test data do not have this character written in capitals. The average score for the generated QA from template MMNR2 is 1.6 and the average score for the pair from template MMNR5 is 2, with "incorrect information" as the major reason for the unacceptable or bad score.

Clause: Playing cards (A0) annoy (V) her (A1) in any way (MNR)
Question (Template MMNR2): Who annoy her in any way ?
Answer (Template MMNR2): playing cards does
Question (Template MMNR5): How does playing cards annoy her?
Answer (Template MMNR5): in any way

Fig. 7.6.: Correct Generated QA Pairs that Got Unacceptable Scores

7.3.3 MLOC

The average score for MLOC is relatively good compared to the results for other categories. However, there is one error that is fixed for the templates in this category. The error is found on the generated QA pairs from the sentence "Inside this house, the Duchess is nursing a pig baby and a cook is having a temper tantrum". Figure 7.7 shows this error.

In a previous improvement, a generalized preposition "in" is added when a location does not have a preposition[2]. But the check misses prepositions that start with an "in". After this improvement, the generated question from template MLOC1 becomes "what does the Duchess do inside this house?" while the generated answer from template MLOC5 becomes "Inside this house".

[2]Such as, to enable a question "what does she meet in the forest?", instead of "What does she meet forest" since the LOC phrase is only "the forest" from the sentence: "She wanders off into the forest, where she meets a Caterpillar sitting on a mushroom and smoking a hookah (i.e., a water pipe)"

Clause: Inside this house (LOC) the Duchess (A0) nursing (V) a pig baby (A1)
Question (Template MLOC1): What does the Duchess do in Inside this house ?
Answer (Template MLOC1): the Duchess nursing a pig baby
Question (Template MLOC5): Where does the Duchess nursing a pig baby ?
Answer (Template MLOC5): in Inside this house

Fig. 7.7.: A Missed Preposition that Causes an Error

There is another generated QA pair that is rated low because it has a strange structure as displayed in Figure 7.8. Having "Alice" as a location can be quite strange. However, since this problem occurs only one time and the average score for the generated QA pairs from this clause is not very low, this problem is not included in the improvement.

Clause: The Hatter (A0) fires (V) riddles (A1) at Alice (LOC)
Question: What happens at Alice ?
Answer: The Hatter fires riddles

Fig. 7.8.: Alice as a Location

7.3.4 MTMP

There is one improvement that is conducted on the MTMP error analysis phase but is applied in other categories as well. Modal modifiers (ArgM MOD) are quite important to be included in the templates because they can change a sentence meaning. Figure 7.9 shows an example from the MTMP category.

Perhaps including the modal "can" to make the generated question become "Whom does she can reach no longer?" does not make a big difference. However, including the ArgM MOD in the templates is still important to keep the correct information of the text. A bigger difference perhaps can be found when the clause is "she cannot reach the key"; a generated question "what does she reach" with a generated answer "she reach the key"

Sentence: she (A0) can (MOD) no longer (TMP) reach (V) the key for it (A1)
Question: Whom does she reach no longer ?
Answer: she reach the key for it

Fig. 7.9.: Including the Modal Modifiers

will not carry the correct information from the text. Therefore, when including the ArgM MOD, the correct information can still be carried by the generated QA pairs, especially the generated answer ("she cannot reach the key" instead of "she reach the key").

Several generated QA pairs that are rated low under the MTMP category have wrong question words. However, they are not fixed because the QAMatcher can still match these questions well. Two of them are as follows:

Sentence: It is not the kind of croquet that Alice is used to, instead of mallets and balls, the Queen's version uses flamingoes and hedgehogs, who become quite unruly when Alice tries to use them
Question: Whom does flamingoes and hedgehogs become when Alice tries to use them ?
Answer: flamingoes and hedgehogs become quite unruly

Sentence: By this time, she has grown again to giantess size, and knocks the jurors flying as she gets up to take the stand
Question: Whom does she grown By this time ?
Answer: she grown to giantess size

7.3.5 ARGU

The improvements that are done for the templates in the ARGU category are all related to syntactic clues. The first one is a check to make sure that objective pronouns are used for objects and subjective pronouns are used for subjects. This check actually has been implemented in other templates but not in ARGU templates. The following example shows

the generated QA pair before the improvement, followed with the generated QA pair from the same source after the improvement.

Sentence: She shrinks again and slips and is swept up by the pool

Question: Who swept up She ?

Answer: by the pool does

After improvement:

Question: Who swept up her ?

Answer: the pool does

The second one is to remove the preposition "by" from answers that consist of "by" as their first word. The following example shows the generated QA pair before the improvement, followed with the generated QA pair from the same source after the improvement.

Sentence: He is interrupted by the sound of the Queen loudly commencing the Knave's trial

Question: Who interrupted He ?

Answer: by the sound of the Queen loudly commencing the Knave s trial does

After improvement:

Question: Who interrupted him ?

Answer: the sound of the Queen loudly commencing the Knave s trial does

7.3.6 DCNJ

Several parsing errors are found under the DCNJ category based on the error analysis of the user evaluation result. Figure 7.10 shows an example of the error. The word "mood" is incorrectly labeled as the root by the PyStanfordDependencies library. Therefore, the template also generates a strange question and answer from this sentence.

Another example is displayed in Figure 7.11. The phrase "with a frog footman outside" is labeled as an "nmod" of the main phrase "she eats some of the...". This causes the information to be incorrectly understood as Alice eats the mushroom with a Frog Footman. The correct dependency tree should have had the phrase "with a frog footman outside"

Sentence: The Duchess is in a terrible mood and rudely addresses Alice before flinging the baby at her

Question: What happens when The Duchess mood ?

Answer: The Duchess mood and rudely addresses Alice before flinging the baby at her

Fig. 7.10.: Dependency Parsing Error - Part 1

under the phrase “a little house”. Since these cases are caused by parsing errors, they are not improved.

Sentence: She eats some of the shrinking side of the mushroom and sees a little house, with a Frog Footman outside, who has received an invitation for the Duchess to attend the Queen of Hearts croquet tournament

Question: What happens when She eats some of the shrinking side of the mushroom with a Frog Footman outside , received ?

Answer: She eats some of the shrinking side of the mushroom with a Frog Footman outside , received and sees a little house

Fig. 7.11.: Dependency Parsing Error - Part 2

8. USER STUDY USING QA MATCHER

The evaluations in chapter 6 and chapter 7 involve a rating scheme to assess the quality of the generated QA pairs. Improvements have been made based on the evaluation results. Next, a user study using the QAMatcher is conducted.

Before conducting the user study, the QAMatcher is prepared by implementing a follow-up question strategy, a solution to resolve the risk of implementing the follow-up question strategy, a pilot evaluation, and an improvement. All of these preparations are explained in section 8.1. Next, the setup of the user study is explained in section 8.2. The user study result and discussion are provided in section 8.3. Finally, the conclusion of the user study is provided in section 8.4.

8.1 Preparing the QAMatcher

8.1.1 Follow-Up Question Strategy

When a person talks to a virtual human, he should be able to say anything to the virtual human and get a relevant response. However, the generated QA pairs from the AQG system are limited to the Alice in Wonderland story that is used as the input. In order to reduce the probability of a user asking questions that are not in the generated QA pairs, a follow-up question strategy is created. A follow-up question strategy here is a strategy that implicitly suggests the user ask a follow-up question that is related to the previous response that the agent gives.

The follow-up question strategy that is implemented for the evaluation is to add the clue "then something happens" and a phrase of the next story piece. This strategy is implemented in 4 QA template categories:

1. **MLOC**: If the next sentence has an ArgM LOC → return the current answer + " Then something happens " + ArgM LOC of the next sentence.

2. **MTMP**: If the next sentence has an ArgM TMP → return the current answer + " Then something happens " + ArgM TMP of the next sentence.

3. **MADV**: If the next sentence has an ArgM ADV → return the current answer + " Then something happens to " + Subj + ArgM ADV of the next sentence.

4. **DCNJ**: If the next sentence has a Conjunction → return the current answer + " Then something happens " + a clause before the conjunction of the next sentence.

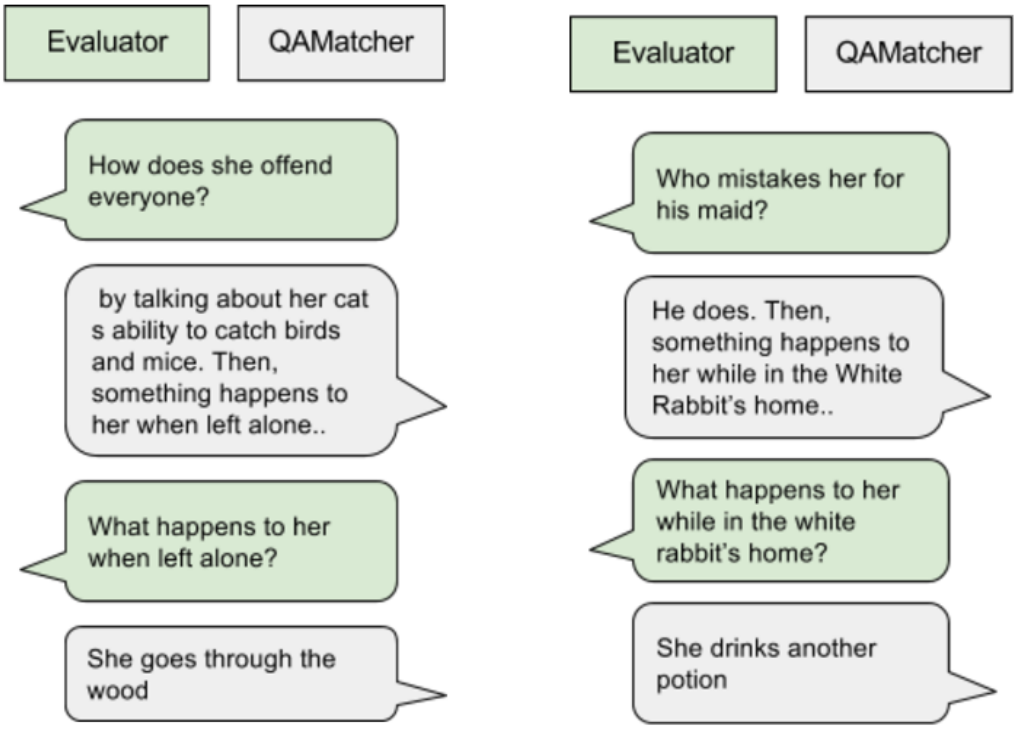

Fig. 8.1.: Examples of the clue implementation for a generated QA in the MADV category

Figure 8.1 shows the implementation of the strategy in two sample dialogues from the MADV category, and Figure 8.2 shows the implementation for the DCNJ category. The second utterance in both figures shows the real answer. The additional "Then something happens..." is implemented in the AQG answer templates as an extension. The third dialogue, "What happens..." is the expected follow-up question that might be asked by the evaluator. This question then can be matched with the QA pairs that are already in the resource.

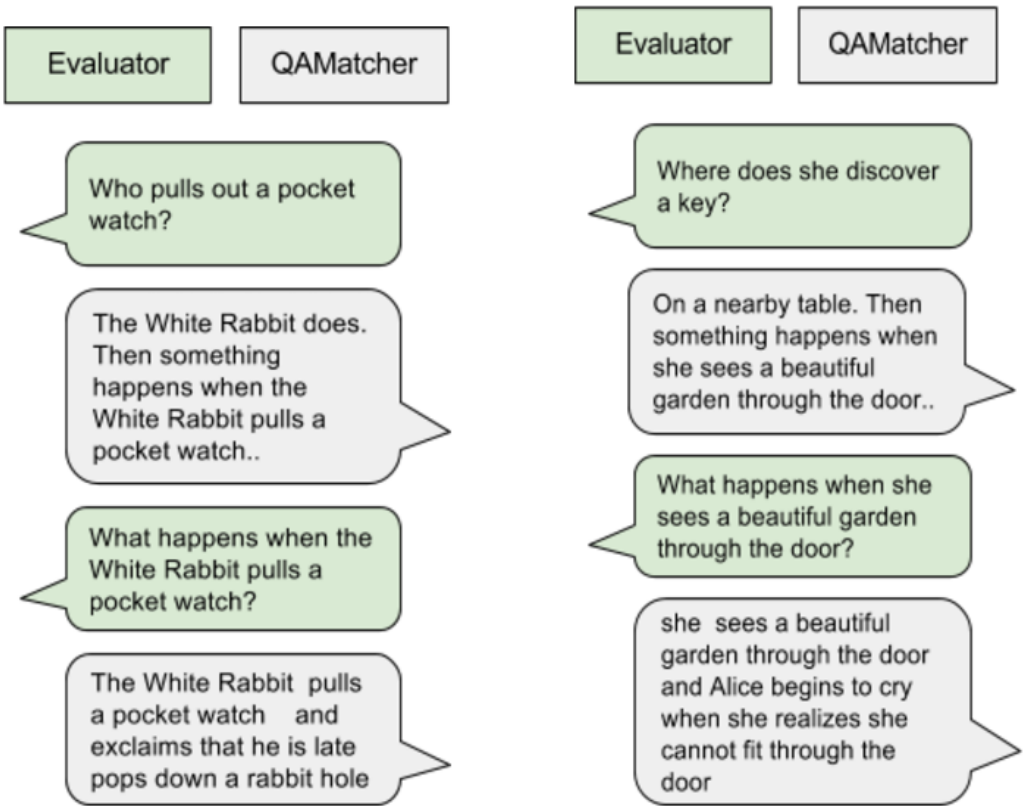

Fig. 8.2.: Examples of the clue implementation for a generated QA in the DCNJ category

The clues are implemented for the MLOC, MTMP, MADV, and DCNJ categories, but not for the MMNR and ARGU categories because MMNR and ARGU do not have a "What happens..." question template. The reason of why they do not have this question template is because it is difficult to separate their elements (Subject, verb, object, modifier) without losing the information from its clause. This means that it is also difficult to come out with a good clue for the evaluators to ask. Beside this reason, it is also good to not have the same "Then something happens..." clue every time the QAMatcher give a response. Other clues might be good to have in the future work.

8.1.2 Risks on the Follow-Up Question Strategy

The follow-up question strategy that is implemented for the second user evaluation using QAMatcher is to add the clue "then something happens" and a phrase of the next story

piece. The positive case for implementing this strategy is that the user will ask a complete follow-up question that can be matched easily to the generated question, such as "what happens when she sees a beautiful garden through the door?", as displayed in Figure 8.2. This question is then matched with the following generated QA pair to give the right answer:

```
<qa category="DCNJ" template="DCNJ1">
 <question>What happens when she  sees a beautiful garden  Through the door
 ?</question>
 <answer>she  sees a beautiful garden  Through the door   and Alice begins
 to cry when she realizes she cannot fit through the door  </answer>
</qa>
```

QAMatcher will still match a user question when it is only a little bit different than the generated question. However, it is also possible that the user will only ask "what happens?" which can be matched with many other generated questions. One solution to resolve this risk is to keep a history of the dialogue between the QAMatcher and the user. Therefore, when the user only asks "What happens", this question can be linked into the previous answer from the QAMatcher, and next, the QAMatcher can give the right answer.

Figure 8.3 illustrates the solution for incomplete follow-up questions using the dialogue example from Figure 8.1. Figure 8.3 shows that the user only asks "What happens" instead of "What happens to her while in the white rabbit's home?". The solution algorithm analyzes whether the user's question is specific enough, by checking if the user starts his question by a phrase "What happens" and followed by 15 or more characters. The user's question "What happens" will not pass the algorithm's check and will go to the "no" option. Therefore, the algorithm will check the last utterance that was given by the QAMatcher, "He does. Then, something happens to her while in the White Rabbit's home...". The phrase after "something happens" will be retrieved by the algorithm and added into the current user's utterance. Therefore, the user's utterance after the process will become "What happens to her while in the White Rabbit's home". The QAMatcher will take the after-process user's utterance and match it with the generated QA pairs. Using the same example, the best generated question that can be matched by the QAMatcher is as follows:

```
<qa category="MADV" template="MADV4">
```

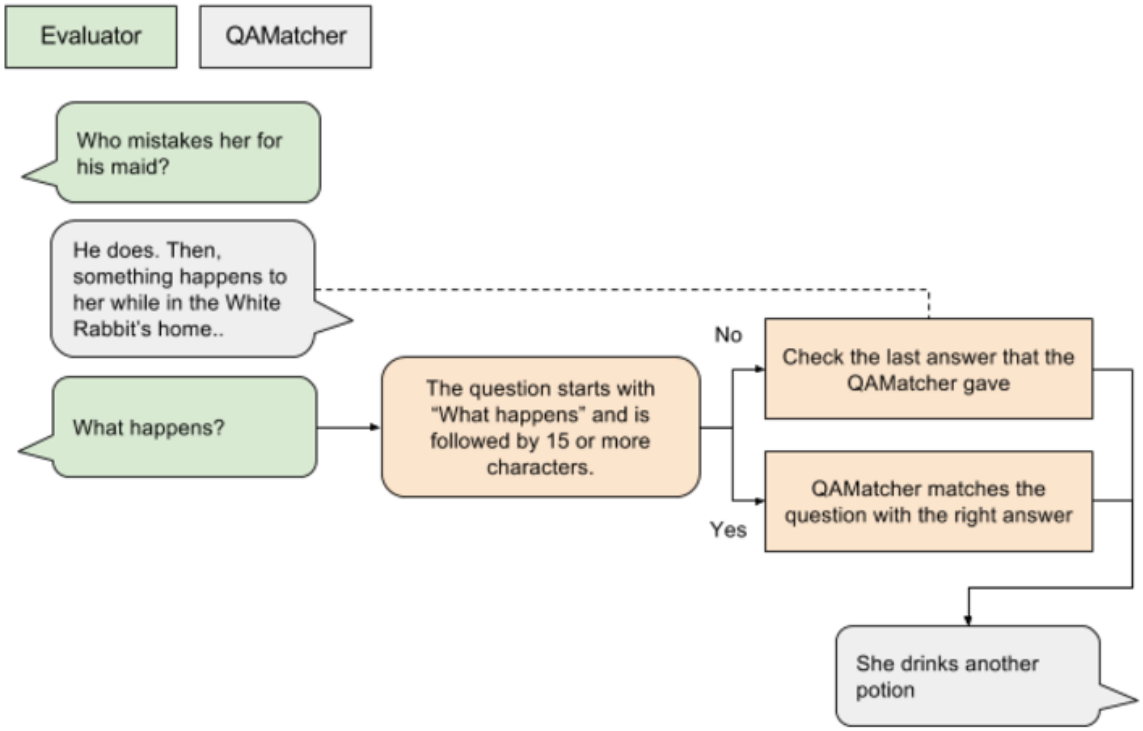

Fig. 8.3.: A solution to resolve the risk of implementing the follow-up question strategy

```
<question>What happens to her While in the White Rabbit s home ?</question>
<answer>she drinks another potion  </answer>
</qa>
```

The incomplete follow-up question solution is implemented in a separate python file named `historysearch.py`, with the dialogue history stored in a text file named `history.txt`. Since the QAMatcher is written in java, a console script is written separately to call both `historysearch.py` and the QAMatcher simultaneously. Therefore, this second user evaluation can be conducted directly by opening the console script.

8.1.3 Pilot Evaluation

A pilot evaluation is conducted by a recently-graduated student from an English taught programme at the University of Twente. The evaluator is able to communicate in English really well. The QA pairs are generated from three summaries that were used previously as

the training and test data; GradeSaver, SparkNotes, and Litcharts. The AQG system that is used has been improved based on the user evaluation (see chapter 7. The pilot evaluator was also involved in the user evaluation, however, his understanding in Alice in Wonderland story is limited.

The pilot evaluator was first interviewed about his knowledge of the Alice in Wonderland story. The interview result was his understanding of the story is limited to a girl who meets a White Rabbit and followed it down a hole, and then meets some strange creatures in Wonderland, without knowing exactly what Alice's experiences are. After being interviewed about his knowledge of the story, the pilot evaluator was told to ask questions about Alice in Wonderland, as he wants to know more about the story. The pilot evaluator was given about 15 to 20 minutes. When the time is finished, he was asked to rate five statements on a 1 to 5 scale. Score 1 is "strongly disagree", score 2 is "disagree", score 3 is "neutral", score 4 is "agree", and score "5" is strongly agree. The statements and rates that the pilot evaluator gave are displayed in Table 8.1. The result of the ratings are not so great. However, the pilot evaluator agreed that he knows more about the story after the evaluation session.

Table 8.1.: The Rating Result for the Post-evaluation Statements of the Pilot Evaluator

No	**Statements**	**Rate**
1	You know more about the story	4
2	The answers make sense	2
3	The answers use correct English	3
4	The dialogue as a whole is coherent	2
5	The dialogue as a whole feels natural	2

The QAMatcher starts with an introduction: "The story begins on an ordinary land, then something happens when Alice was sitting with her sister outdoors...". Next, the evaluator can ask anything to the QAMatcher. Figure 8.4 shows the dialogue that between the evaluator and the QAMatcher.

A1	The story begins on an ordinary land, then something happens when Alice was sitting with her sister outdoors...
U1	What is the name of alice's sister?
A2	Alice sitting with her sister outdoors when she spies a White Rabbit with a pocket watch . Then something happens to her when Fascinated by the sight ...
U2	What happens when alice is fascinated by the sight?
A3	she follows the rabbit down the hole . Then something happens when She falls for a long time ...
U3	What does she fall on?
A4	She falls for a long time and finds herself in a long hallway full of doors

Fig. 8.4.: The dialogue between the pilot evaluator (U) and the QAMatcher (A) - Part 1

Overall, the follow-up question strategy works well with the pilot evaluator. He almost always responds back according to the answers which implement the strategy. For example, the first question that was asked by the pilot evaluator (U1) is "What is the name of Alice's sister?". This question is related to the information that was given in the follow-up strategy phrase "then something happens when Alice was sitting with her sister outdoors..." (A1). However, this follow-up question is not the intended question. The second question from the pilot evaluator (U2), however, is the intended follow-up question from the strategy. It asks about the information that was given in the phrase "then something happens to her when fascinated by the sight..." (A2). The QAMatcher matches this question (U2) well and gives the correct answer (A3).

The incomplete follow-up question also happens several times during the evaluation. For example, the follow-up question that the pilot evaluator asked, "what happens?" as shown on the utterance U23 and U24 in Figure 8.5. However, the solution to resolve the risk of this incomplete follow-up question works well and gives the right answer as shown on the utterances A24 and A25 respectively.

However, there is one major issue that appears on this pilot evaluation. The answers always seem to follow the story despite the questions that are asked. This problem occurs only when the previous answer that was given by the QAMatcher consists of the follow-up

U21	what does the caterpillar do?
A22	The Caterpillar and Alice get into an argument but before the Caterpillar crawls away in disgust , he tells Alice that different parts of the mushroom will make her grow or shrink . Then something happens when Alice tastes a part of the mushroom ...
U22	what happens when she tastes the mushroom?
A23	she spies the door lined hallway from the beginning of her adventures . Then something happens when This time she prepared ...
U23	what happens?
A24	This time she prepared and she manages to get to the lovely garden that she saw earlier . Then something happens when She walks ...
U24	what happens?
A25	She walks and finds herself in the garden of the Queen of Hearts

Fig. 8.5.: The dialogue between the pilot evaluator (U) and the QAMatcher (A) - Part 2

strategy "then something happens...". This is apparently caused by the proposed solution to resolve the risk of an incomplete follow-up question. The solution, as illustrated on Figure 8.3, is to give an answer based on the previous answer if the users do not ask a complete question such as "what happens when she tastes a part of the mushroom". This solution is quite naive and can cause problems when the users want to ask other questions. This also prevents the process to evaluate the relevancy of an answer given a question. However, this might be the reason why the pilot evaluator agreed with the statement "You know more about the story".

8.1.4 Improvement

Based on the pilot evaluation, the solution to face the risk of incomplete follow-up question is improved. By using several regular expressions, the follow-up questions that will be linked to the previous QAMatcher's answers are around:

- What happen(s)(ed)(?)
- What happen(s)(ed) next(then)(after)(?)

- What('s) next(then)(does)(after)(?)
- What(?)
- And then(next)(?)
- Then(next)(?)

The improved solution is also based on the follow-up question observation from the pilot evaluation; there are 5 "What does?", 4 "What happens?", and 1 "What?". After improving the solution to resolve the incomplete follow-up question, the evaluation is further conducted with 4 people.

8.2 User Study Setup

The user evaluation using the QAMatcher is conducted with 4 evaluators. All of the evaluators are students or recently graduated students from the University of Twente. The evaluators are not native English speakers but they understand English well and regularly speak English in their daily lives.

The first and the fourth evaluator were not involved in the first user evaluation of rating the generated QA pairs. The second and third evaluator were involved in the evaluation of rating the generated QA pairs. However, all of the evaluators were first asked about their knowledge about Alice in Wonderland story, and all of them have a very limited knowledge about it. They only know that the story is about a girl who experienced strange events in the Wonderland.

The first, second, and third evaluators conduct the evaluation without being told about the summary of the story in the beginning. The fourth evaluator is, however, first explained about the summary of Alice in Wonderland from Wikipedia in order to see if the result of the evaluation can be better when an evaluator is given a brief summary beforehand. All of the evaluators then are told to ask any questions about Alice in Wonderland as they want to know more about the story. They are given about 15 to 20 minutes. When the evaluation finishes, they are given 5 post-evaluation statements that they need to rate in a 1 to 5 scale (the same procedure with the pilot evaluation in subsection 8.1.3).

The data for this user study is the same as the one that is used for the pilot user study. They are the training data of AQG and the test data from previous evaluations (chapter 6 and chapter 7); the summaries from GradeSaver, NoteSparks, and Litcharts. The three of them generate 691 QA pairs from 172 sentences.

8.3 User Study Result and Discussion

After the evaluation session with the QAMatcher, the evaluators are asked to rate 5 statements using 5-scale scheme. Score 1 for "strongly disagree", 2 for "disagree", 3 for "neutral", 4 for "agree", and 5 for "strongly agree". Figure 8.2 displays the score, with A as the first evaluator, B as the second evaluator, C as the third evaluator, and D as the fourth evaluator.

Table 8.2.: The Rating Result for the Post-evaluation Statements of the Four Evaluators

Statement	**A**	**B**	**C**	**D**	**Average**
You know more about the story	4	3	2	2	2.75
The answers make sense	3	3	2	2	2.5
The answers use correct English	3	4	4	3	3.5
The dialogue as a whole is coherent	3	3	1	2	2.25
The dialogue as a whole feels natural	2	2	2	2	2

The number of relevant answers (including their percentage based on the number of questions asked by the evaluator) that the QAMatcher gave for the evaluators are provided in Table 8.3. The number of relevant but incorrect answers are also provided in the table because a virtual human is expected to give relevant answers, however, correct answers might be desirable too.

Table 8.3.: The Number of Relevant Answers

Evaluator	Number of Questions	Number of Relevant Answers	Number of Relevant but Incorrect Answers
A	51	18 (35%)	9
B	34	10 (29%)	5
C	34	11 (32%)	2
D	62	19 (30%)	4

8.3.1 Result from the First Evaluator

The first evaluator almost always asks question following the previous answer from the QAMatcher. The follow-up question strategy works well because most of them ask about the information that consists in the strategy phrase "then something happens...". However, the evaluator's follow-up questions are about something else that does not exist in the story. An example is displayed in Figure 8.6.

A2	Alice does. Then something happens when The King of Hearts interrupts their conversation ...
U2	what is the King's name?
A3	the note does. Then something happens when The Queen becomes furious with Alice and orders her beheading ...
U3	is she happy with the King?
A4	the Cheshire Cat does. Then something happens when the game abandoned ...

Fig. 8.6.: The dialogue between the 1st evaluator (U) and the QAMatcher (A) - 1

Overall the questions that are asked do not get correct answers. Most of them because the answers are not in the story or the source input. However, there are several questions that are supposed to be in the source input, but not in the generated QA pair. An example is displayed in Figure 8.7.

A4	the Cheshire Cat does. Then something happens when the game abandoned ...
U4	what kind of game is it?
A4	she goes through the wood . Then something happens when He mistakes her for his maid ...

Fig. 8.7.: The dialogue between the 1st evaluator (U) and the QAMatcher (A) - 2

The game information is actually written in the source input:

> The Queen takes Alice to join in the croquet game. It is not the kind of croquet that Alice is used to, instead of mallets and balls, the Queen's version uses flamingoes and hedgehogs, who become quite unruly when Alice tries to use them. Also, nobody takes turns, so the pitch is suddenly a mess with animals and playing cards. The Queen gets very irate, calling for mass executions. Meanwhile, the Cheshire Cat has returned and is causing trouble with the King, but when the Queen's officers try to catch him, he vanishes. So the game is abandoned and the Queen turns her attention to Alice.

The first sentence from the story above perhaps can give the information about the game in the simplest way. However, the generated questions from this sentence are different. They are "Who takes Alice?" and "Who join in the croquet game?", instead of telling that croquet is the kind of game that is played.

There is also a question that asks about the information that is given as the follow-up question strategy that does not have a correct answer to it. This question is already good by including "What happens" on the beginning, however, the QAMatcher cannot match it well with the right answer because the question is not specific enough. Despite the incorrect answer that is given, the answer is still quite coherent with the question asked. Which means that the QAMatcher seems to understand the question and give an answer accordingly. This question is shown in Figure 8.8.

The question U32 is actually supposed to be matched with the following QA pair:

```
<qa category="DCNJ" template="DCNJ1">
 <question>What happens when she  sees a beautiful garden  Through the door
```

A32	she discovers a key . Then something happens when she sees a beautiful garden Through the door ...
U32	What happens in the garden?
A33	she meets a Caterpillar sitting on a mushroom and smoking a hookah (i.e. , a water pipe) . Then something happens when The Caterpillar and Alice get into an argument ...

Fig. 8.8.: The dialogue between the 1st evaluator (U) and the QAMatcher (A) - 3

```
 ?</question>
 <answer>she  sees a beautiful garden  Through the door   and Alice begins
 to cry when she realizes she cannot fit through the door  </answer>
</qa>
```

However, the question U32 is matched with a wrong QA pair instead:

```
<qa category="MLOC" template="MLOC4">
 <question>What happens in the forest ?</question>
 <answer>she meets a Caterpillar sitting on a mushroom and smoking a hookah
 ( i.e. , a water pipe ) . Then something happens when The Caterpillar and
 Alice  get into an argument  ...</answer>
</qa>
```

The first evaluator asks a few questions that match with the follow-up question strategy and its solution to handle the risk of the incomplete follow-up question. These questions are successfully matched with the relevant answers. The dialogue is displayed in Figure 8.9.

8.3.2 Result from the Second Evaluator

Similar with the first evaluator, the second evaluator asks several questions about things that are not in the story (e.g. “What is the flavor of the cake?”, “What is the color of the flowerpot?”). However, the second evaluator notices that the program cannot give the right answers to that and start to ask questions about the phrase that is part of the follow-up question strategy (Figure 8.10).

A11	she meets a Caterpillar sitting on a mushroom and smoking a hookah (i.e. , a water pipe) . Then something happens when The Caterpillar and Alice get into an argument ...
U11	What happen there?
A12	The Caterpillar and Alice get into an argument but before the Caterpillar crawls away in disgust , he tells Alice that different parts of the mushroom will make her grow or shrink . Then something happens when Alice tastes a part of the mushroom ...

A17	The first time . Then something happens when she pokes her head into the branches of a tree ...
U17	What happen then?
A18	she pokes her head into the branches of a tree and meets a Pigeon

Fig. 8.9.: The dialogue between the 1st evaluator (U) and the QAMatcher (A) - 4

A24	She follows the rabbit down the hole . Then something happens when She falls for a long time ...
U24	What happens after she falls for a long time?
A25	She falls for a long time and finds herself in a long hallway full of doors

A39	the Duchess does. Then something happens when the Duchess nursing a pig baby Inside this house ...
U39	What happens when the Duchess nursing a pig baby?
A40	the Duchess nursing a pig baby Inside this house and a cook is having a temper tantrum

Fig. 8.10.: The dialogue between the 2nd evaluator (U) and the QAMatcher (A) - 1

There are several questions about information that exist in the story and are asked by the second evaluator but they do not get correct answers. The utterance U48 and A49 on Figure 8.11 show that a question that does not have the answer in the generated QA pairs might get a response that is incorrect and irrelevant. The question U49 and answer A50 are not very good as well but at least the answer is still related to the Knave of Hearts.

U48	What does the Chesire Cat do?
A49	the Gryphon drags Alice soon afterward . Then something happens when The Knave of Hearts accused ...
U49	What is the Knave of Hearts accused of?
A50	The Knave of Hearts accused but the evidence against him is very bad

Fig. 8.11.: The dialogue between the 2nd evaluator (U) and the QAMatcher (A) - 2

There is no QA pair generated about the things that are asked in Figure 8.11 because both input sentences only have clauses that fall under the ARGU category, which only has one template (ARGU2) that asks about the subject. It seems better to include the template ARGU1 and ARGU3 which have been excluded based on the pre-initial evaluation (explained in section 6.1). When only the subject that is mentioned in the question, such as "What does Alice do?", there can be several answers linked to this question. The user might ask about what Alice does in the first chapter and the QAMatcher responds with what Alice does in the second chapter. However, for a virtual human, it is better to respond to this question with a relevant but incorrect answer, rather than an incorrect and irrelevant answer. An example of a relevant but incorrect answer is when a user asks "What does Alice do" and refer to the story when Alice is in the White Rabbit's house. However, the answer that the user gets is "Alice finds a pig baby", which refers to the story when Alice is at the house of the Duchess. When the template ARGU1 and ARGU3 are included, the following QA pairs should be generated.

```
<sentence word="The Knave of Hearts has been accused of stealing the tarts
of the Queen of Hearts , but the evidence against him is very bad ">
 <qa category="ARGU" template="ARGU3">
  <question>What is The Knave of Hearts accused?</question>
  <answer>The Knave of Hearts is accused of stealing the tarts of the Queen
  of Hearts </answer>
 </qa>
</sentence>
<sentence word="The Cheshire Cat explains to Alice that everyone in
```

```
Wonderland is mad , including Alice herself ">
 <qa category="ARGU" template="ARGU1">
  <question>What does The Cheshire Cat  do?</question>
  <answer>The Cheshire Cat  explains that everyone in Wonderland is mad ,
  including Alice herself . Then something happens when The Cheshire Cat
  gives directions  to the March Hare s house  ...</answer>
 </qa>
</sentence>
```

8.3.3 Result from the Third Evaluator

There are more relevant answers given for the third evaluator. This is perhaps because the third evaluator rarely asks questions that are not in the story. There are several questions that do not follow the follow-up question strategy but they still get relevant and even correct answers. Figure 8.12 displays an example of it.

U6	What animal sitting on the mushroom?
A7	a Caterpillar does

Fig. 8.12.: The dialogue between the 3rd evaluator (U) and the QAMatcher (A) - 1

U14	Does she meet someone when she is stretched out?
A15	she pokes her head into the branches of a tree

Fig. 8.13.: The dialogue between the 3rd evaluator (U) and the QAMatcher (A) - 2

However, there are several answers that are not correctly given but still related to the question, such as the question and answer displayed in Figure 8.13. Perhaps the question "Does she meet someone when she is stretched out?" (U14) would have been better if the it was matched this question with the following QA pair in the QAMatcher:

```
<qa category="DCNJ" template="DCNJ1">
```

```
 <question>What happens when she  pokes her head  into the branches of a
 tree  ?</question>
 <answer>she  pokes her head  into the branches of a tree   and meets a
 Pigeon  </answer>
</qa>
```

However, the user's question (U14) was matched with the following QA pair instead:

```
<qa category="MADV" template="MADV1">
 <question>What does she do While stretched out ?</question>
 <answer>she pokes her head  into the branches of a tree </answer>
</qa>
```

The answer "she pokes her head into the branches of a tree" perhaps is not so relevant but it is still related because in the story, she pokes her head into the branches of a tree and then meets a pigeon.

There is one time when the third evaluator seems to ask about whom does the pronoun refer to, as shown in utterance U12 on Figure 8.14. The answer that is given is actually quite relevant. However, since the current AQG system does not include pronoun resolution, the answer that is given still uses the pronoun "she" instead of using the person's name, Alice.

A12	she uses them The first time . Then something happens when she pokes her head into the branches of a tree ...
U12	who pokes her head?
A7	she pokes her head into the branches of a tree

Fig. 8.14.: The dialogue between the 3rd evaluator (U) and the QAMatcher (A) - 3

There are several questions that follow the follow-up strategy as well, as shown in Figure 8.15. These questions successfully give relevant and correct answers.

Despite the overall better result, incorrect and irrelevant answers are still given for the third evaluator. Figure 8.16 shows an example of it. What the user meant to ask is perhaps about the pocket watch that the White Rabbit has. However, this is not explained explicitly

A30	the Cheshire Cat does. Then something happens when The King takes offense ...
U30	What happen when The King takes offense?
A31	The King takes offense but arranges for the Cheshire Cat s execution since the Cheshire Cat is now only a head floating in midair , no one can agree on how to behead it . Then something happens when The Duchess approaches Alice and attempts ...

Fig. 8.15.: The dialogue between the 3rd evaluator (U) and the QAMatcher (A) - 4

U30	What does white rabbit bring all the time?
A12	she uses them The first time . Then something happens when she pokes her head into the branches of a tree ...

Fig. 8.16.: The dialogue between the 3rd evaluator (U) and the QAMatcher (A) - 5

in the summaries that are used as the test data. It is only explained that there is a White Rabbit with a pocket watch or is pulling a pocket watch.

There is one question that is supposed to get a correct answer based on the completeness of the utterance, as shown in Figure 8.17. This question is almost similar as the generated question that is stored in the resource of the QAMatcher. However, apparently the spaces between the words can make the QAMatcher incorrectly match the questions. Figure 8.18 shows how these spaces create problems on the generated answers on the QAMatcher.

U13	What happened after she pokes her head into the branches of a tree?
A14	While she is stretched out

Fig. 8.17.: The dialogue between the 3rd evaluator (U) and the QAMatcher (A) - 6

8.3.4 Result from the Fourth Evaluator

Since the fourth evaluator is first told about the summary of Alice in Wonderland, the evaluator asks several questions about some questions that do not follow the previous answer

```
Enter question: what happened when she  pokes her head  into the branches of a tree?
she  pokes her head  into the branches of a tree   and meets a Pigeon
Enter question: what happened when she pokes her head into the branches of a tree?
While she is stretched out
Enter question:
```

Fig. 8.18.: A difference in spaces can give a different answer

or the follow-up question strategy phrase from the QAMatcher. Two sample questions are displayed on Figure 8.19. These questions that are shown in utterance U8 and U19 are not mentioned in the dialogue before.

A8	when she is Left alone . Then something happens when He mistakes her for his maid ...
U8	who is hatter

A19	on the tabletop above her head
U19	who drink tea with Alice in the tea party?

Fig. 8.19.: The dialogue between the 4th evaluator (U) and the QAMatcher (A) - 1

A different problem that can be noticed from the fourth evaluator result is that it is also necessary to link a follow-up question such as the one shown by the utterance U43 on Figure 8.20. The follow-up question “then” (U43) cannot retrieve the next piece of the story because the previous utterance (A43) does not have the follow-up question strategy phrase (“then something happens...”). When these follow-up question phrases (e.g. “then?”, “what’s next?”) are handled for all questions that the user asks and not only when the previous response from the QAMatcher has a follow-up question strategy phrase, then the QAMatcher can keep telling the next story to the user.

8.4 User Study Conclusion

In conclusion, the current generated QA pairs from the AQG system cannot be used by themselves for the QAMatcher. There were more irrelevant answers than relevant ones that

A43	Alice tastes a part of the mushroom and her neck stretches above the trees
U43	then?
A44	Alice sitting with her sister outdoors when she spies a White Rabbit with a pocket watch . Then something happens to her when Fascinated by the sight ...
U44	then?
A45	she follows the rabbit down the hole . Then something happens when She falls for a long time ...

Fig. 8.20.: The dialogue between the 4th evaluator (U) and the QAMatcher (A) - 2

were given by the QAMatcher in the user study. The post-evaluation statement ratings were also not great.

The AQG system might be used to provide the domain knowledge of Alice. However, various question templates that ask about the same thing are necessary in order for the QAMatcher to match the user's questions and the prepared questions better. Tools that provide synonyms might be necessary to create more varied templates.

The follow-up question strategy is a good solution to give the users some ideas of what to ask, and also to keep the users to ask about things that have answers. The history of the dialogue between the user and the QAMatcher is important to be kept for the current follow-up strategy, and for other purposes in the future work.

9. CONCLUSION AND FUTURE WORK

In this chapter, the summary of this research is provided first, and later the conclusion and the future work are discussed.

9.1 Summary

The introduction chapter explains about the ARIA-VALUSPA project which develops a framework of virtual humans. One work package that is being developed at the University of Twente is called Multi-Modal Dialogue Management for Information Retrieval. The work package is conducted for a virtual human called Alice, who is a representation of the character Alice in the Alice in Wonderland story. One challenge in developing multi-modal dialogue management for information retrieval is preparing the domain knowledge for the virtual human. Question generation is chosen as an approach to create the domain knowledge of Alice.

The concepts of conversational agents and virtual humans are also explained, including dialogue management which is a module in conversational agents or virtual humans that is responsible to manage the dialogue between Alice and its users. Next, the dialogue manager of Alice is explained together with the domain knowledge that is managed by a tool called QAMatcher. The QAMatcher works by matching a user's question with existing questions by using text processing algorithms. The existing questions in the QAMatcher will be generated by the chosen approach, question generation. The topic of the questions that will be generated focuses on Alice in Wonderland related story.

Question generation is a subject in natural language processing that intends to generate questions from text. Question generation is usually used for helping teachers to make questions for their students. However, recent research show that question generation can also be used for other purposes such as internet security domain and virtual humans. There are two main approaches in conducting question generation research, they are the syntactic approach and the semantic approach. Syntactic approach usually explores the use of syntactic

tools such as Stanford Parser and Tregex while the semantic approach explores the semantic tools such as Stanford Dependency and Semantic Role Labels (SRL). The approach that is chosen for the Alice Question Generation system is the semantic approach.

Alice Question Generation (AQG) is a question generation system that is developed to generate question and answer pairs about Alice in Wonderland. AQG uses SRL as the main task to retrieve the semantic meaning of Alice in Wonderland story. SENNA is the SRL tool that is used to retrieve the SRL. Beside SRL, AQG also uses Stanford Dependency to retrieve the semantic meaning of Alice in Wonderland story. PyStanfordDependencies is the Stanford Dependency tool that is used for the AQG system.

The first phase in building the AQG system is observing the SRL patterns based on the frequency of the pattern occurrences and the consistency of the semantic information conveyed by the pattern across different sentences. Two summaries of Alice in Wonderland from GradeSaver and SparkNotes are used as the training data. Based on the observation, the pattern that consist of 2 or more Arguments, 0 or more Modifiers, and 1 verb, is chosen to be included in the AQG templates. The modifiers that are chosen are adverbials, manners, locatives, and temporals. Next, the dependency labels are observed for sentences that have conjunctions, because sentences with conjunctions are most likely separated into different clauses by SENNA and can lose a complete information from the sentence.

The second phase in building the AQG system is creating the templates. The template creation focuses on the events (actions, happenings) and existents (characters, settings). The questions in the templates ask about the subject, the predicate, and the object of the events and existents. In the initial version of the AQG, there are 25 templates of question and answer pairs that fall under 6 categories. Next, initial evaluation of the templates is conducted by the author. The templates that create too many errors are removed from the AQG system, and other templates are improved. After initial evaluation, there are 19 templates under 6 categories that are included in the system.

The AQG system is next evaluated by 6 annotators by using a 5-scale rating system. The test data is a summary of Alice in Wonderland from Litchart that consists of 69 sentences. The annotators are divided into two groups. The first group consists of 3 annotators and 35 sentences, while the second group consists of 3 annotators and 34 sentences. The first group evaluate 137 question and answer pairs from the 35 sentences, while the second group

evaluate 131 pairs from the 34 sentences. The average score from both groups 3.495 out of 5. A last improvement on the template is conducted before the next evaluation with the QAMatcher.

The QAMatcher is first set up by a follow-up question strategy in order the keep the evaluators to ask questions about Alice in Wonderland only. A pilot evaluation is first conducted and the follow-up strategy is improved. Next, the user study is conducted with 4 evaluators. The evaluators are given about 15 to 20 minutes time to ask about Alice in Wonderland as they want to know more about the story. The result from this user study is that there were more irrelevant answers than relevant ones that were given by the QAMatcher. The current generated QA pairs from the AQG system cannot be used by themselves for the QAMatcher. More varied templates that ask about the same thing are necessary to be created in the future work. The follow-up question strategy and the history of the dialogue between the user and the QAMatcher are important implementation for the current user study and for other purposes in the future work.

9.2 Conclusion and Future Work

In conclusion, the current generated QA pairs from the AQG system cannot be used by themselves for the QAMatcher because there were more irrelevant answers than relevant ones that were given by the QAMatcher in the user study. Furthermore, there are several things that are important to be considered when implementing QG systems for virtual humans. They are discussed in subsection 9.2.1. The discussions on conducting the user study is provided in subsection 9.2.2.

9.2.1 Automatic Question Generation for Virtual Humans

First, the question is not the only important part of a question generation (QG) system. Answers are also important because they are the ones that the virtual human shows to the users. It is different compared to QG systems that are used for teaching since they usually focus more on the question formulation. QG for virtual humans needs a good question formulation by considering its performance when being processed by text algorithms, and

it also needs a good answer formulation by considering its naturalness when virtual humans give this answer to the user.

A recent research of QG focuses on the importance of information in a sentence pattern before generating questions [28]. This approach is proven to overcome the result of prior works in QG which focus on creating as many questions as possible. QG for virtual humans, however, performs better when there are more generated questions. A way that it can be done is to create a variety of questions that ask about the same thing. For example, an answer "Alice follows the rabbit to a rabbit hole" can have several questions such as "Where does Alice follow the Rabbit?", "Where does Alice go when she sees the rabbit?", "What does Alice do when she sees the rabbit?". A challenge on creating various questions is, however, to have a good semantic parsing result that is consistent across different sentences. In the current implementation, one question is always paired with one answer. However, more varied questions can be created in the future work.

Most QG researchers evaluate their systems by having annotators rate the generated questions. This is somehow more difficult to be conducted as it is with QG for virtual humans. Most annotators prefer a simpler answer to a question. For example, the question "Where does Alice follow the rabbit?" is better to have a direct answer "to a rabbit hole" than a more complete answer "Alice follows the rabbit to a rabbit hole". In virtual humans, however, having direct answers every time can make the virtual humans appear less natural. In future work, it can be better to have direct answers for the evaluation that rates the generated question and answer and more complete phrases can be implemented for the answers when conducting the user study with QA matching tools. The another way is to change the evaluation procedure instead.

More complete phrases are also preferred for the user study with QA matching tools such as QAMatcher. When having a more comprehensive answer, several questions can be matched with one complete answer. Figure 9.1 shows an answer that is more comprehensive. When the answer is direct, the probability that the QAMatcher gives relevant answers might decreases. Figure 9.2 shows the expected utterances when the answers are direct. When the user asks "where does Alice follow the rabbit" (Question 1) there might be a chance that the QAMatcher mistakenly matches this question with "Alice follows the rabbit" (Answer 2) because there are many similar words in both questions. When the answer is more

comprehensive as shown in Figure 9.1, question 1 and 2 from the Figure 9.2 can be matched with the same answer "Alice follows the rabbit to a rabbit hole".

Question 1: Where does Alice follow the rabbit?

Question 2: What does Alice do when she sees the rabbit?

Answer: Alice follows the rabbit to a rabbit hole

Fig. 9.1.: QA pairs that use a comprehensive answer

Question 1: Where does Alice follow the rabbit?

Answer 1: to a rabbit hole

Question 2: What does Alice do when she sees the rabbit?

Answer 2: Alice follows the rabbit

Fig. 9.2.: QA pairs that use more direct answers

9.2.2 User Study using QA Matcher

A follow-up question strategy and a history file of the dialogue between the user and the QAMatcher are important implementations for conducting a user study with the QA-Matcher. The follow-up question strategy can give the users some ideas of what to ask. It also keeps them to ask about things that have the answers. The history file is important for this follow-up question strategy, and also other purposes such as giving the next piece of the story every time the user wants to know what happens next, or to refer pronouns that are mentioned in the previous utterances. Pronoun resolution also needs to be implemented in the future work.

The history file is also important for the QAMatcher to keep track of what has been discussed. This is especially important to know which part of the story the question asks about. Another way to keep the context of the conversation together is to separate the domain knowledge into several story pieces. For example, by keeping the chapters' numbers behind the generated QA pairs and knowing which chapter the user asked about can already keep the context of the story. This strategy, however, cannot be dependent only on the QG

system, because most of the semantic tools only take one sentence as the input and it somehow already misses the context. The strategy of keeping the context of the story can be implemented in the future work.

REFERENCES

REFERENCES

[1] X. Yao, E. Tosch, G. Chen, E. Nouri, R. Artstein, A. Leuski, K. Sagae, and D. Traum, "Creating conversational characters using question generation tools," *Dialogue & Discourse*, vol. 3, no. 2, pp. 125–146, 2012.

[2] D. Jurafsky and J. H. Martin, *Speech and Language Processing (2Nd Edition)*. Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 2009.

[3] A. L. Gorin, G. Riccardi, and J. H. Wright, "How may i help you?" *Speech communication*, vol. 23, no. 1, pp. 113–127, 1997.

[4] W. R. Swartout, J. Gratch, R. W. Hill Jr, E. Hovy, S. Marsella, J. Rickel, D. Traum *et al.*, "Toward virtual humans," *AI Magazine*, vol. 27, no. 2, p. 96, 2006.

[5] W. Swartout, D. Traum, R. Artstein, D. Noren, P. Debevec, K. Bronnenkant, J. Williams, A. Leuski, S. Narayanan, D. Piepol *et al.*, "Ada and grace: Toward realistic and engaging virtual museum guides," in *International Conference on Intelligent Virtual Agents*. Springer, 2010, pp. 286–300.

[6] D. DeVault, R. Artstein, G. Benn, T. Dey, E. Fast, A. Gainer, K. Georgila, J. Gratch, A. Hartholt, M. Lhommet *et al.*, "Simsensei kiosk: A virtual human interviewer for healthcare decision support," in *Proceedings of the 2014 international conference on Autonomous agents and multi-agent systems*. International Foundation for Autonomous Agents and Multiagent Systems, 2014, pp. 1061–1068.

[7] A. Hartholt, D. Traum, S. C. Marsella, A. Shapiro, G. Stratou, A. Leuski, L.-P. Morency, and J. Gratch, "All together now," in *International Workshop on Intelligent Virtual Agents*. Springer, 2013, pp. 368–381.

[8] M. Schröder, E. Bevacqua, F. Eyben, H. Gunes, M. ter Maat, S. Pammi, E. de Sevin, M. Valstar, and M. Wöllmer, "Final sal system. project deliverable d1d, semaine," 2010.

[9] ARIA-VALUSPA, "Deliverable d3.2: Implementation of adaptive task-based dialogue system," 2016.

[10] M. ter Maat and D. Heylen, "Flipper: An information state component for spoken dialogue systems," in *International Workshop on Intelligent Virtual Agents*. Springer, 2011, pp. 470–472.

[11] J. Wagner, F. Lingenfelser, and E. André, "The social signal interpretation framework (ssi) for real time signal processing and recognition." in *INTERSPEECH*, 2011, pp. 3245–3248.

[12] H. Bunt, J. Alexandersson, J. Carletta, J.-W. Choe, A. C. Fang, K. Hasida, K. Lee, V. Petukhova, A. Popescu-Belis, L. Romary *et al.*, "Towards an iso standard for dialogue act annotation," in *Seventh conference on International Language Resources and Evaluation (LREC'10)*, 2010.

[13] N.-T. Le, T. Kojiri, N. Pinkwart *et al.*, "Automatic question generation for educational applications-the state of art." in *ICCSAMA*. Springer, 2014, pp. 325–338.

[14] M. Liu, R. A. Calvo, and V. Rus, "G-asks: An intelligent automatic question generation system for academic writing support," *Dialogue & Discourse*, vol. 3, no. 2, pp. 101–124, 2012.

[15] M. Heilman and N. A. Smith, "Question generation via overgenerating transformations and ranking," CARNEGIE-MELLON UNIV PITTSBURGH PA LANGUAGE TECHNOLOGIES INST, Tech. Rep., 2009.

[16] ——, "Extracting simplified statements for factual question generation," in *Proceedings of QG2010: The Third Workshop on Question Generation*, 2010, p. 11.

[17] ——, "Good question! statistical ranking for question generation," in *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*. Association for Computational Linguistics, 2010, pp. 609–617.

[18] M. Heilman, "Automatic factual question generation from text," Ph.D. dissertation, Carnegie Mellon University, 2011.

[19] K. Mazidi and R. D. Nielsen, "Pedagogical evaluation of automatically generated questions," in *International Conference on Intelligent Tutoring Systems*. Springer, 2014, pp. 294–299.

[20] S. S. Woo, Z. Li, and J. Mirkovic, "Good automatic authentication question generation." in *INLG*, 2016, pp. 203–206.

[21] A. Leuski and D. R. Traum, "Npceditor: A tool for building question-answering characters." in *LREC*, 2010.

[22] N. Schlaefer, P. Gieselmann, and G. Sautter, "The ephyra qa system at trec 2006," in *Proceedings of the Fifteenth Text REtrieval Conference, 2006*, 2006.

[23] K. Mazidi and R. D. Nielsen, "Leveraging multiple views of text for automatic question generation," in *International Conference on Artificial Intelligence in Education*. Springer, 2015, pp. 257–266.

[24] ——, "Linguistic considerations in automatic question generation." in *ACL (2)*, 2014, pp. 321–326.

[25] M. P. Marcus, M. A. Marcinkiewicz, and B. Santorini, "Building a large annotated corpus of english: The penn treebank," *Computational linguistics*, vol. 19, no. 2, pp. 313–330, 1993.

[26] O. Babko-Malaya, "Propbank annotation guidelines," *URL: http://verbs. colorado. edu*, 2005.

[27] M.-C. De Marneffe, B. MacCartney, C. D. Manning *et al.*, "Generating typed dependency parses from phrase structure parses," in *Proceedings of LREC*, vol. 6, no. 2006. Genoa Italy, 2006, pp. 449–454.

[28] K. Mazidi and P. Tarau, "Infusing nlu into automatic question generation," in *The 9th International Natural Language Generation conference*, 2016, p. 51.

[29] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa, "Natural language processing (almost) from scratch," *Journal of Machine Learning Research*, vol. 12, no. Aug, pp. 2493–2537, 2011.

[30] S. B. Chatman, *Story and discourse: Narrative structure in fiction and film.* Cornell University Press, 1980.

APPENDICES

A. APPENDIX: ALICE QUESTION GENERATION

Table A.1.: Initial QA Templates

No	Template	Template Structure
1	MADV1 *(Asks about the predicate)*	Q: What + aux + lower Arg + do to + higher Arg + ArgM ADV + ? A: lower Arg + V + higher Arg
2	MADV2 *(Asks about the subject)*	Q: Who + V + higher Arg + ArgM ADV + ? A: lower Arg
3	MADV3 *(Asks about the object)*	Q: What + aux + lower Arg + V + ArgM ADV + ? A: lower Arg + V + higher Arg
4	MADV4 *(Asks about the predicate and) the object)*	Q: What happens to + lower Arg + ArgM ADV + ? A: lower Arg + V + higher Arg
5	MADV5 *(Asks about the modifier adverbial)*	Q: When + aux + lower Arg + V + higher Arg + ? A: ArgM ADV
6	MMNR1 *(Asks about the predicate)*	Q: What + aux + lower Arg + do to + higher Arg + ArgM MNR + ? A: lower Arg + V + higher Arg
7	MMNR2 *(Asks about the subject)*	Q: Who + V + higher Arg + ArgM MNR + ? A: lower Arg

continued on next page

Table A.1.: *continued*

No	Template	Template Structure
8	MMNR3 *(Asks about the object)*	Q: What + aux + lower Arg + V + ArgM MNR + ? A: lower Arg + V + higher Arg
9	MMNR4 *(Asks about the predicate and) the object)*	Q: What happens to + lower Arg + ArgM MNR + ? A: lower Arg + V + higher Arg
10	MMNR5 *(Asks about the modifier manner)*	Q: How + aux + lower Arg + V + higher Arg + ? A: ArgM MNR
11	MLOC1 *(Asks about the predicate)*	Q: What + aux + lower Arg + do to + higher Arg + ArgM LOC + ? A: lower Arg + V + higher Arg
12	MLOC2 *(Asks about the subject)*	Q: Who + V + higher Arg + ArgM LOC + ? A: lower Arg
13	MLOC3 *(Asks about the object)*	Q: What + aux + lower Arg + V + ArgM LOC + ? A: lower Arg + V + higher Arg
14	MLOC4 *(Asks about the predicate and) the object)*	Q: What happens to + lower Arg + ArgM LOC + ? A: lower Arg + V + higher Arg
15	MLOC5 *(Asks about the modifier location)*	Q: Where + aux + lower Arg + V + higher Arg + ? A: ArgM LOC

continued on next page

Table A.1.: *continued*

No	Template	Template Structure
16	MTMP1 *(Asks about the predicate)*	Q: What + aux + lower Arg + do + higher Arg + ArgM TMP + ? A: lower Arg + V + higher Arg
17	MTMP2 *(Asks about the subject)*	Q: Who + V + higher Arg + ArgM TMP + ? A: lower Arg
18	MTMP3 *(Asks about the object)*	Q: What + aux + lower Arg + V + ArgM TMP + ? A: lower Arg + V + higher Arg
19	MTMP4 *(Asks about the predicate and) the object)*	Q: What happens to + lower Arg + ArgM TMP + ? A: lower Arg + V + higher Arg
20	MTMP5 *(Asks about the modifier location)*	Q: When + aux + lower Arg + V + higher Arg + ? A: ArgM TMP
21	ARGU1 *(Asks about the predicate)*	Q: What + aux + lower Arg + do to + higher Arg + ? A: lower Arg + V + higher Arg
22	ARGU2 *(Asks about the subject)*	Q: Who + V + higher Arg + ? A: lower Arg
23	ARGU3 *(Asks about the object)*	Q: What + aux + lower Arg + V + ? A: lower Arg + V + higher Arg

continued on next page

Table A.1.: *continued*

No	Template	Template Structure
24	ARGU4 *(Asks about the predicate and) the object)*	Q: What happens to + lower Arg + ? A: lower Arg + V + higher Arg
25	DCNJ1 *(Asks about the story from other) clauses)*	Q: What happens when + Subj + V + Dobj + Nmod + ? A: Subj + V + Dobj + Nmod + Conj

Table A.2.: QA Templates After Improvement

No	Template	Template Structure
1	MADV1 *(Asks about the predicate)*	Q: What + aux + lower Arg + do + ArgM ADV + ? A: lower Arg + V + higher Arg + higher Args
2	MADV2 *(Asks about the subject)*	Q: Who + V + higher Arg + ArgM ADV + ? A: lower Arg + V + higher Arg + higher Args
3	MADV3 *(Asks about the object)*	Q: What is it that + lower Arg + V + ArgM ADV + ? A: lower Arg + V + higher Arg + higher Args
4	MADV4 *(Asks about the predicate and) the object)*	Q: What happens to + lower Arg + ArgM ADV + ? A: lower Arg + V + higher Arg + higher Args
5	MADV5 *(Asks about the*	Q: When + aux + lower Arg + V + higher Arg + higher Args + ?

continued on next page

Table A.2.: *continued*

No	Template	Template Structure
	modifier adverbial)	A: ArgM ADV
6	MMNR2 *(Asks about the subject)*	Q: Who + V + higher Arg + ArgM MNR + ? A: lower Arg + aux
7	MMNR5 *(Asks about the modifier manner)*	Q: How + aux + lower Arg + V + higher Arg + ? A: ArgM MNR
8	MLOC1 *(Asks about the predicate)*	Q: What + aux + lower Arg + do + ArgM LOC + ? A: lower Arg + V + higher Arg
9	MLOC2 *(Asks about the subject)*	Q: Who + V + higher Arg + ArgM LOC + ? A: lower Arg + aux
10	MLOC3 *(Asks about the object)*	Q: What + aux + lower Arg + V + ArgM LOC + ? A: lower Arg + V + higher Arg
11	MLOC4 *(Asks about the predicate and) the object)*	Q: What happens + ArgM LOC + ? A: lower Arg + V + higher Arg
12	MLOC5 *(Asks about the modifier location)*	Q: Where + aux + lower Arg + V + higher Arg + ? A: ArgM LOC
13	MTMP1 *(Asks about the predicate)*	Q: What + aux + lower Arg + do + ArgM TMP + ? A: lower Arg + V + higher Arg

continued on next page

Table A.2.: *continued*

No	**Template**	**Template Structure**
14	MTMP2 *(Asks about the subject)*	Q: Who + V + higher Arg A: lower Arg + V + higher Arg + ArgM TMP
15	MTMP3 *(Asks about the object)*	Q: Whom + aux + lower Arg + V + ArgM TMP + ? A: lower Arg + V + higher Arg
16	MTMP4 *(Asks about the predicate and) the object)*	Q: What happens + ArgM TMP + ? A: lower Arg + V + higher Arg
17	MTMP5 *(Asks about the modifier location)*	Q: When + aux + lower Arg + V + higher Arg + ? A: ArgM TMP
18	ARGU2 *(Asks about the subject)*	Q: Who + V + higher Arg + higher Args + ? A: lower Arg + aux
19	DCNJ1 *(Asks about the story from other) clauses)*	Q: What happens when + Subj + V + Ccomps + Xcomps + Dobj + Nmod + ? A: Subj + V + Ccomps + Xcomps + Dobj + Nmod + Conj

B. APPENDIX: USER EVALUATION

B.1 Instruction for Question and Answer Rating

Instruction

The goal of this evaluation is to rate question and answer (Q&A) pairs according to their acceptability for literal reading, which means that the Q&A pairs are for basic understanding for the facts that are presented in the text, and not about the implications, characters' feelings, conclusions, etc. The Q&A pairs in this evaluation are intended to be carried by a person and a virtual human. The virtual human is called Alice, who is the representation of the character Alice from the Alice in Wonderland story. Therefore, the Q&A that will be exchanged between the person and Alice is about Alice in Wonderland.

The question is the representation of the question that a person can ask to Alice, while the answer is the representation of the answer that Alice can respond back to the person. The question and the answer are not to be rated separately. Therefore, when the question is good but the answer is strange, then a "good score" cannot be assigned to this pair and vice versa.

The focus of the evaluation is the awkwardness and the information that is being carried away by the Q&A pairs. Grammars and pronoun references are not the focus of this evaluation, unless the grammars or the pronoun references create difficulties in understanding the Q&A, then it can be categorized in the Awkwardness/Other problem. Please read the sentence before rating the Q&A pair. Next, rate the Q&A pair according to the 5-scale score which is described below. When score 2 or 1 is given, please give either "Incorrect Information" or "Awkwardness/Other", or both. Finally, please rate each Q&A pair independently. For example, if there are two similar Q&A pairs, please rate them the same even though they seem redundant.

Scoring Explanation	
Good (5)	This Q&A pair does not have any problems, and it is a good as the one that a person might ask and the virtual human might answer.
Acceptable (4)	The Q&A does not have any problems.
Borderline (3)	The Q&A might have a problem, but I'm not sure.
Unacceptable (2)	The Q&A definitely has a minor problem.
Bad (1)	The Q&A has major problems.

Fig. B.1.: 5-Scale Scoring System

Reason Explanation (only for score 2 and 1)	
Incorrect Information (a)	The Q&A implies something that is obviously incorrect according to the context.
Awkwardness/Other (b)	The Q&A is awkwardly phrased or has some other problem (e.g., no native speaker of English would say it this way, the question word is wrong).

Fig. B.2.: Explanation of Unacceptable or Bad Score Reason

No	Sentence	Question	Answer	Score	Reason	Comment
1	While in the White Rabbit s home , she drinks another potion and becomes too huge to get out through the door	When does she drinks another potion ?	While she is in the White Rabbit s home	5		This Q&A pair does not have any problems, and it is a good as the one that a person might ask and the virtual human might answer.
2	He gives her some valuable advice , as well as a valuable tool : the two sides of the mushroom , which can make Alice grow larger and smaller as she wishes	Who gives some valuable advice , as well as a valuable tool : the two sides of the mushroom , which can make Alice grow larger and smaller as she wishes her?	He does	4		This Q&A pair does not have any problems, but the question seems to be too detailed and long for a person to ask.
3	Alice goes to the March Hare s house , where she is treated to a Mad Tea Party	Who goes to the March Hare s house , where she is treated to a Mad Tea Party ?	Alice does	3		The question mentions "where she is treated..." which makes it pretty obvious that the answer is Alice. Therefore, this Q&A pair might have a problem. But I'm not sure.
4	There is later a cake with a note that tells her to eat ; Alice uses both , but she cannot seem to get a handle on things , and is always either too large to get through the door or too small to reach the key	Who get a handle on things ?	she does	2	a	The correct information is Alice cannot seem to get a handle on things, therefore, a question "who get a handle on things" is incorrect according to the given context.
5	She longs to get there , but the door is too small	What happens when She longs get ?	She longs get but the door is too small	1	b	This Q&A pair has major problem because the question structure is strange that it is difficult to understand

Fig. B.3.: Example of rated Q&A pairs