



HOKKAIDO UNIVERSITY

Title	A Study on Dialogue Agent Adapting to Various Situations
Author(s)	Jordan, Arnaud
Degree Grantor	北海道大学
Degree Name	博士(情報科学)
Dissertation Number	甲第12191号
Issue Date	2016-03-24
DOI	https://doi.org/10.14943/doctoral.k12191
Doc URL	https://hdl.handle.net/2115/61751
Type	doctoral thesis
File Information	Jordan_Arnau.pdf



学位論文

Doctoral Dissertation

A Study on Dialogue Agent Adapting to Various Situations

(多様な状況に適応する対話エージェントに関する研究)



北海道大学
HOKKAIDO UNIVERSITY

北海道大学大学院情報科学研究科

Jordan Arnaud

Thesis Abstract

In the present world, machines and computers are used everywhere all days long by many different users. Therefore, many different kinds of interface have been set up in order to communicate with machines, such as for example buttons, mouse, touchscreen or keyboard. On the other hand, humans used to use natural languages such as Japanese or French in order to communicate each other. That is why it is commonly accepted that for humans it is more natural to speak, use gestures and so forth to communicate. In this context, interfaces using natural languages gained considerable attention in recent years. Especially with the democratization of smartphones, voice interfaces and dialogue agents attract even more attention. However, even if speech recognition precision increased continuously, computers are still not able to fully understand all the aspects of natural languages, and as a result users feel they are hard to talk and to use. Moreover, speech recognition precision already reached a ceiling and cannot be largely improved, as a result, the malaise felt by the user is considered due to the deficiency of the responses generated by the system more than to voice processing. That is why in order to solve this problem I consider new interactive processing have to be developed to achieve natural conversation between humans and machines.

In order to achieve a smooth dialogue with humans, several faculties such as casual chatting, question answering, emotional reaction, joke or gesture are required. As a first step, I considered developing a dialogue agent focusing on casual dialogue to handle daily life conversation. Concretely, the proposed agent does not achieve particular task, but focuses on providing natural responses corresponding to the user input. The present thesis proposes several innovative methods based on the human's behavior and presents the results of the developed dialogue agents.

For many years, diverse dialogue agents have been developed. However, they generally use a limited static knowledge in order to give priority to experiment results. Contrary, the present thesis does not only focus on the systems' results, but also on the future possible developments of the system and attempts to propose flexible and adaptive systems. Concretely, the research converges on dialogue agents constantly adapting to the environment where they are used. Therefore, several methods and dialogue agents are proposed. In addition, experiments have been carried out to prove their reliability and to discuss their advantages and drawbacks.

The present thesis is composed of four chapters. In chapter one, the introduction presents the background of the research and the state of the art of dialogue agents. In chapter two, as a first step, a dialogue agent handling question answering for casual dialogue in Japanese and the related experimental performance evaluation are presented. Namely, the proposed method uses rule acquisition and interrogative pronoun identification to generate responses.

This method uses a morphological analyzer to identify interrogative pronouns present in the input sentence, and then selects the most adequate word among all the knowledge

it acquired to substitute it, and then generate the output. In addition, several methods of output generation using knowledge acquired from different users are presented, tested and compared. Moreover, in order to facilitate the use of the developed systems several interfaces have been developed such as a web page and smartphone applications providing vocal dialogue.

However, the proposed systems depend on the language. Furthermore, implementing all the possible inputs or input structures using this kind of language-limited methods is complicated. Especially, the system cannot be enough flexible to handle unknown words or linguistic phenomena since predicting all the inputs is impossible. That is why the proposed first methods are considered insufficient to achieve the target precision.

In chapter three, the feasibility of language-independent dialogue agent, i.e. a system which does not limit the language of the input, is studied and a method is proposed. Concretely, the proposed method accepts any language as input. Contrary to methods which require huge external resources and tools, the proposed method attempts to limit them as much as possible in order to focus on improving the flexibility and the knowledge acquisition processing and finally obtain an improvement of the precision of the system. Therefore, dictionaries and morphological analyzers are not used since they are language-dependent. Basically, the newly proposed dialogue agent represents all the knowledge it acquires during dialogues into a graph, and uses the latter to generate outputs. As a consequence, previously proposed graph algorithm can be adapted to the system. For example, results prove that the proposed method can infer new knowledge using graph clustering. In addition, parallel simultaneous graph traversal has been efficiently implemented in the system due to the hierarchical construction of the graph. Concretely, the proposed dialogue agent represents all the acquired knowledge into a graph and accepts any kind of input at any time, i.e. even during the output generation, and then generates one or several outputs in function of its internal state. Experiments in five languages; Chinese, English, French, Korean and Japanese has been carried out and the results exceed a more relative multilingual dialogue agent. Moreover, since the proposed method handles any kind of input, a system upgrade enabling emotional input was proposed and a result improvement compared to the previous agent was observed. Therefore, since language-independent methods can handle any tag representing emotion or environment similarly to textual input, they are considered well-adapted to the development of multimodal dialogue agent.

Chapter four concludes the present thesis. Especially, details about the research progress, the evolution of the objectives and the final results are presented. In addition, advantages and drawbacks of each method are discussed and possible improvements are proposed. Moreover, future evolutions and challenges of dialogue agents, as well as new objectives are broached.

学位論文内容の要旨

近年、機械やコンピュータは様々なユーザに至る所で利用されている。したがって、機械を操作するため、状況に応じ、ボタンやマウス、キーボード、タッチスクリーンなど多種多様なインタフェースを設置する必要がある。一方、人間は、日本語やフランス語などの自然言語を利用して、コミュニケーションを行う。そのため、音声やジェスチャーなどを用いたコミュニケーションが、人間にとって最も自然だと考えられる。このような観点から、自然言語を用いたインタフェースの研究が注目を集めてきた。さらに、スマートフォンの普及によって、音声インタフェースや対話エージェントがより重視されるようになった。しかし、音声認識の精度がいくら向上しても、コンピュータは依然として人間の言葉を完全に理解することができないので、ユーザは、音声インタフェースに対して親しみにくく、使いづらいといった印象を持ってしまう。また、現在の音声認識はすでに高精度であり、精度面での大幅な向上は期待できない。ユーザが感じる違和感は、音声処理の問題ではなく、主に対話処理によりコンピュータが生成する応答文の内容の不十分さが原因であると考えられる。したがって、この問題を解決するためには、人間との自然な対話を可能とする対話処理技術の開発が必要であると考えられる。

機械が人間と円滑にコミュニケーションを行うためには、雑談や質問応答、情動反応、冗談、ジェスチャーなど様々な能力が必要である。著者は、研究の第1歩として、雑談対話エージェントに焦点を当て、人間と自然な日常会話を行うことができるエージェントの構築を目的として研究を行った。具体的には、特定のタスクは行わず、人間らしい応答をするエージェントを開発することである。本学位論文では、人間の態度に基づいた新規性の高い手法を提案し、対話エージェントの開発を行った結果について述べる。

これまでに多様な対話システムが開発されてきたが、実験結果を優先し、限られた知識に基づいて構築されていることが多い。本研究では、論文における手法の実験のみに集中するのではなく、エージェントの今後の発展も視野に入れ、柔軟性と適応性の高い手法を目指した。すなわち、使用状況に常に適応する対話エージェントの仕組みについての研究を行った。また、様々な手法を提案し、複数の対話エージェントの開発を行った。さらに、それらの実用性を確かめるため、実験を行い、各仕組みのメリットとデメリットについての検討を行った。

本論文は4章から構成される。第1章の序論では、研究の背景と既存の関連研究について述べた。第2章では、第一段階として、日本語のみに絞った雑談における質問応答システムについて構築を行い、その性能評価実験の結果の検討を行った。具体的には、応答ルール獲得と疑問代名詞の認識を用いた手法の開発を行った。

この手法は、形態素解析器を使用して、入力文に含まれた疑問代名詞を求める。次に、その時点までに獲得したルールにより選択された適切な単語にその代名詞を置き換え、応答文を生成する。さらに、応答文を生成する複数の手法を提案し、ユーザから知識を獲得し、その知識を他のユーザが利用する実験を行った。実験ではそれらの手法の有効性を比較した。また、対話システムの利用をより容易にするため、ウェブページやスマートフォンのアプリなど、様々なデバイスを用いた音声対話を含む対話インタフェースの構築を行った。

しかしながら、この段階では手法が言語に依存したものとなっていた。このように一つの言語に限定されたシステムでは、対話におけるすべての構文や入力の種類などを予測することが困難であった。さらに、ユーザのすべての入力を事前に網羅することは不可能なため、未知の単語や言語現象に対処するのに必要な柔軟性が不足しているため、現状のシステムでは高い精度に到達することが困難であると考えられた。

第3章では、対象言語を限定した手法に柔軟性を持たせるため、対象言語に限定しない、すなわち言語に依存しない対話エージェントの実現可能性の調査を行い、手法の検討を行った結果について述べている。具体的には、どのような言語でも対処することができる手法の提案を行った。本手法は、膨大な外部資源やツールを利用するシステムと異なり、システムの精度の向上を図ると同時に外部ツールへの依存をできる限り少なくすることにより、柔軟性および学習効率の向上を目指した。したがって、言語に依存する辞書や形態素解析器などは利用していない。つまり、新たに提案したエージェントは、対話から獲得したあらゆる知識をグラフにより表し、そのグラフのみ利用して応答文を生成する。グラフを用いることにより、既に提案されたグラフのアルゴリズムを利用することができる。例えば、グラフを用いてクラスタリングを行い、新たな知識を生成することが可能であり、そのことの確認を行った。その上、グラフの階層構造を利用することにより、並列計算を用いたグラフ探索を容易に行うことができ、リアルタイムのプログラムを実装することが可能となった。具体的には、開発したエージェントは対話の中から獲得した知識をグラフで表し、応答文生成中でも常にあらゆる言語の入力ができ、エージェントの内部状況に応じて一つまたは複数の応答文を出力する。日本語、中国語、韓国語、英語、フランス語の5言語で実験を行い、従来の言語に依存する多言語の対話システムを上回る応答精度を得た。さらに、言語に依存しないエージェントはどのような入力も可能であることを利用し、文字列のみではなく感情入力を可能としたエージェントの開発を行った。感情入力のない状態と比較実験を行うことで、感情入力による応答文の精度の向上を確認した。このことから、言語に依存しない手法は、感情や環境を表したタグを、言葉を表す文字列と同等に処理できるので、マルチモーダル対話システムの実現において有効なものと考えられる。

第4章では、結論について述べた。具体的には、対話エージェントに関する本研究の進捗や目標の変化、研究結果について詳細に述べた。また、提案した各手法のメリットとデメリットを説明し、それらへの対処方法の紹介を行った。さらに、対話エージェントの今後の改良および将来の展望をまとめ、本研究の結果に基づいた新たな研究目的についても述べた。

Contents

1	Introduction	1
1.1	Dialogue agents	1
1.2	Objectives of the research	2
1.3	State of the art	3
1.3.1	A limited world	3
1.3.2	Rule-based systems	4
1.3.3	Inductive learning	4
1.3.4	Using the user feedback	5
1.3.5	Inductive learning method based on genetic algorithm with sexual selection	5
1.3.6	Statistical systems	6
1.3.7	Main problems	6
2	Language-specific agent	8
2.1	Introduction	8
2.2	External resources	9
2.2.1	Dictionaries	10
2.2.2	Morphological analysis	10
2.2.3	Lexical tools	11
2.2.4	Syntactic analysis	11
2.3	Rule-based generic system	11
2.3.1	System architecture	12
2.3.2	Behaviour	20
2.3.3	Knowledge's sharing	22
2.3.4	Saving	22
2.3.5	Interfaces	22
2.3.6	Experiments	26
2.3.7	Conclusion	30
2.4	Knowledge sharing	31
2.4.1	System architecture	31
2.4.2	Two different methods to share knowledge	32
2.4.3	Experiments	34
2.4.4	Conclusion	40

2.5	Conclusion	40
3	Language-independent agent	42
3.1	Introduction	42
3.1.1	Objectives	44
3.2	Technical constraints	44
3.3	Real-time language-independent dialogue agent	45
3.3.1	System architecture	45
3.3.2	Definition of the graph	47
3.3.3	Used links	48
3.3.4	Graph generation	51
3.3.5	New node inferring	52
3.3.6	Link characteristics and node score calculation	53
3.3.7	Output generation	54
3.3.8	Implementation of the system	58
3.3.9	Training samples	66
3.3.10	Experiments	68
3.3.11	Conclusion	74
3.4	Implementation of emotional tags	75
3.4.1	Links	76
3.4.2	Link characteristics	79
3.4.3	Forgetting process	79
3.4.4	Trigger value	80
3.4.5	Emotional processing	81
3.4.6	Starting knowledge	83
3.4.7	Experiments	84
3.4.8	Conclusion	88
3.5	Conclusion	88
3.6	Future works	89
4	Conclusion	91
4.1	Conclusion	91
4.2	Future works	92
	Bibliography	93
	Acknowledgements	100
	Research achievements	101

List of Figures

1.1	Example of two sentences crossover	5
1.2	Example of a sentence mutation	6
2.1	Example of external resources utilization	9
2.2	Example of definition for the English word “funny”	10
2.3	Example of definition for the Japanese word “楽しい”	10
2.4	System’s data flow	13
2.5	Example of sentence levels	14
2.6	Example of a structure of the rules	15
2.7	Simple rule data flow	15
2.8	Example of a simple rule	15
2.9	Rule-based system’s data flow	16
2.10	Abstract rule data flow	16
2.11	Recursive rule data flow	17
2.12	Example of a recursive rule output	17
2.13	Example of rules arrangement	19
2.14	Example of <i>Aiduchi</i>	21
2.15	Response of a declaration	21
2.16	Response to a question	21
2.17	System’s interface on Android	23
2.18	System’s interface on iOS	24
2.19	Web interface of the system	25
2.20	Robot used as interface of the system	26
2.21	Experiment dialogue example	28
2.22	Semantic differential experiment’s results	29
2.23	Processing of user’s input	32
2.24	Example of a dialogue for SAMU	32
2.25	Example of a dialogue for SAEU	33
2.26	A dialogue sample for SAMU	35
2.27	A dialogue sample for SAEU	35
2.28	A dialogue sample for ELIZA	35
2.29	A dialogue sample for a human	36
2.30	Experiment interface	37
2.31	Semantic differential experiment’s results	39

3.1	Language information flowchart	45
3.2	Principle of output generation	46
3.3	Flowchart of the system	47
3.4	Example of a simple graph	48
3.5	Example of a graph containing different substitution links	50
3.6	Example of two possible clustering results	51
3.7	Flowchart of the score update	55
3.8	Flowchart of the output selection	56
3.9	Processing of user's input	57
3.10	Example of handling of misspelling	57
3.11	System's threads hierarchy	58
3.12	Algorithm used to generate association links	59
3.13	Algorithm used to infer new sentence	60
3.14	Algorithm used when the user inputs a sentence	60
3.15	Algorithm used to check whether a node can split another	61
3.16	Algorithm used to add a new sub-node to the graph	61
3.17	Algorithm used to create substitution links	62
3.18	Algorithm used to find nodes related to the input	63
3.19	Algorithm used to update score of the node	63
3.20	Algorithm used to generate output	64
3.21	Example of trigger changes	65
3.22	Example of a dialogue sample	67
3.23	Different ways to ask one's age in different languages	67
3.24	Example of dialogue with the proposed system	72
3.25	Example of a graph containing different substitution links	78
3.26	Algorithm used to check links which have to be forgotten	80
3.27	Example of graph containing emotion nodes	81
3.28	Example of simple dialogue	83
3.29	Example of emotion sample	84
3.30	Example of a dialogue with the system	85

List of Tables

2.1	Rules characteristics	20
2.2	Subject's information	27
2.3	Results of the preliminary questionnaire	27
2.4	Semantic differential experiments results	28
2.5	Preparation phase subject's information	34
2.6	Subject's information	37
2.7	Experiment's result	38
2.8	Evaluation of the answers	39
3.1	Link characteristics	54
3.2	Threads period	59
3.3	Subject's information	71
3.4	Corpus information	71
3.5	Speech recognition precision	72
3.6	Experiment results	73
3.7	Link characteristics	79
3.8	Corpus information	84
3.9	Subject's information	85
3.10	Speech recognition precision	86
3.11	Number of emotion	86
3.12	Input emotion	86
3.13	Experiments results	87

Chapter 1

Introduction

1.1 Dialogue agents

Nowadays, machines and computers¹ are present everywhere, most of them are driven by specific software adapted to them and their use. The main objective of this software is to control the machine hardware in function of different inputs and make the system as efficient as possible, i.e. reducing the cost of the task for which it is made. One of the most important input is the user command. However, communicating with a machine is still considered complicated and not intuitive by most of users. In order to solve this problem, many human interfaces² have been developed and can be used to indicate what the user wants to be done by the machine. For example, buttons, mouses and keyboards are commonly used. Moreover, for some years, especially in the case of smart-phone, the precision of speech interfaces increased and it is possible to give some simple instructions by voice, using for example *Siri*³ on *iPhone* or *Google Now*⁴ on *Android* devices. However, it is still not possible to speak to a computer like to a human in order to make it do something or to simply have an enjoyable casual chatting. That is why in order to solve this problem and to make the communication with machines easier, it is necessary to make machines able to understand the natural language used by the user, and to react to it in a human way.

Since computers exist, making them able to understand humans was a natural objective and a popular research field. Since the *ELIZA system* [1] has been developed in 1966, a lot of dialogue agents have been proposed [2, 3, 4] or are still under development. Some of them focus on casual chatting, while others like the *Watson system* [5] focus on providing information or other on achieving a particular task such as opening a specific application or sending message to someone.

However, it is still not possible to control or to enjoy a dialogue in natural language with a computer. This is because conversations are a very complicated process, a casual

¹Computer, smartphone, industrial robot, etc...

²Man-machine interface (MMI).

³<http://www.apple.com/ios/siri/>

⁴<https://www.google.com/landing/now/>

conversation is composed of many different aspects such as greeting, simple question, joke and emotion. In addition, there are many different expressions to express the same concepts; humans often use grammatically erroneous sentences and many information are omitted since they can be inferred from the context of the conversation. Moreover, everybody understand a word a little differently and the meaning of words evolves with the time, for example the meaning of the word “nice” used to mean “silly, foolish, simple” while now it is a compliment. That is why I consider a good dialogue agent has to be very flexible and be able to evolve in function of the user language. It has to be able to acquire new knowledge and be able to infer new words meanings in function of previously acquired words. Concretely, I consider a good dialogue agent has to acquire language similarly as humans do in order to learn all the different aspects of a dialogue. Especially, not only words but also other kinds of input have to be acquired by the agent to provide a full lively dialogue.

In addition, dialogue agents are commonly split in two main categories: *task-oriented system* and *non-task-oriented system*. The first focuses on achieving some specific tasks, such as opening an application or providing information to the user. While the second focuses on providing a natural response to the user in order to handle an enjoyable casual conversation. In this research, I considered casual conversation provided by non-task-oriented systems are important in order to acquire information about the user, such as his personal information or the expressions he often uses. Then this information can be used to improve the precision of a task-oriented system. For example, if during a casual dialogue the user said “I like Chinese food” and then sometime after he asks “Show me good restaurants in Tokyo”, the system has to be able to give more importance to “Chinese restaurant” in its results in order to fit the user tastes. Therefore, I consider *non-task-oriented* and *task-oriented methods* have to be used together to combine their advantages and improve their precision.

That is why during my research, I focused on *non-task-oriented dialogues*, since I considered them as the first step to build a complete system which at the end may be able to handle both task and non-task oriented dialogues at the same time. In addition, one of the most important point of this research was to create a flexible dialogue agent which can acquire knowledge automatically and which is able to answer simple questions as well as casual chatting.

1.2 Objectives of the research

The final aim of the dialogue agent’s research field is to make a system able to learn and to speak any language like a human. Especially, one of the objectives is to make systems more human-like [6], and as consequence, indiscernible from humans and passing the *Turing test* [7]. However, this objective is still very far to be reached and as a consequence most of previously proposed system only focused on a part of dialogues, such as questions answering or casual chatting.

During my research I also had the same process, consisting in selecting a kind of dialogue, then developing a system which can handle it. Progressively, the research objectives

changed to a more generic system, and I created new algorithms able to reach these new constraints. The main objective of this research was to proposing new innovative generic algorithms which are considered required to reach a new level of humanity in dialogue processing.

In this thesis, I explain in details each proposed dialogue agent. In addition, I also describe the evolution of the research objectives and the consequence on the developed systems. Finally, I discuss the last proposed system and future possible evolution of the research.

In **Section 1.3**, I make a state of the art of main kinds of dialogue agent. Then, in **Chapter 2**, I present my first system [8] which was language-dependent and focused only on casual chatting in Japanese. In **Chapter 3**, I explain the research final objective which was developing a language-independent algorithm [9] to handle a simple conversation in any language without any code modification or adaptation. Finally, I conclude in **Chapter 4** on the impact of the research’s results and on the future possible development.

1.3 State of the art

Since dialogue agents have been studied for many years, various methods to generate and to output a sentence in function of an input have been proposed. In this section, the principal methods are listed.

1.3.1 A limited world

Handling all the complexity of a natural language required a lot of knowledge. That is why some researchers chose to firstly limit their system to a very simple world such as the *SHRDLU system* [10]. Concretely, the system can only understand elements which are present in its limited world. The world is built using a set of definitions of the different terms understandable by the system. These definitions can have various parameters, but in general all the concepts which have ambiguity such as imaginary things are excluded since they are too complicated to be defined using simple logical rules. In addition, responses to each possible case can also be defined using simple rules. For example, if all the terms of the input are compatible, the system can be set up to output “It is correct”.

Considering the system knows all the elements present in the input and all their parameters, it can proceed to a complete analyze and output a correct response using previously defined sentences. For example, if the two elements “cat” and “dog” are defined with the respective parameters (“do not like”, “dog”) and (“do not like”, “cat”), then if the user inputs “Do cats like dogs?” the system can answer “No”⁵. Since the two parameters are not compatible. However, if the question is “Do cats like mouse?”, the system will not be able to output a correct sentence since the system does not know the meaning of the word “mouse”.

This method works well for very simple world, however since in the reality there are many particular cases such as contradictions or evolutions during time, this kind of method

⁵“cats” and “dogs” can be processed as “cat” and “dog” using a simple plural forms dictionary.

cannot be used to achieve the objective of casual chatting, since the user is supposed to be able to input any sentence he wants. In addition, contrary to this method, humans do not need to understand and to know perfectly all the meaning of an utterance to attempt answer it, since they can make hypotheses and approximations.

1.3.2 Rule-based systems

One of the most intuitive methods to create a dialogue agent is to create a *rule* containing an output for each possible input. For example, a rule can be set to output “I am fine” when the input is “How are you?”. This method is very simple and efficient to create a system when the inputs are limited and the dialogues are short. That is why for example, one of the first dialogue agent, ELIZA is *rule-based*.

In addition, it is possible to use generic rules, such as “Do you like <noun>?” to match a maximum of inputs of the user. Especially, modern rule-based systems use regular expressions that make them very efficient and intuitive to develop. Moreover, the system can also select rules which are the closest to the input even if it does not perfectly match, in order to be able to answer any input of the user. In order to facilitate the administration of all these rules can be set in specific file such as AIML [11] file which are used by the *ALICE chat-bot* [12]. Nevertheless, rule-based dialogue agents are static and always reply the same output each time the same sentence is input.

Therefore, this method is not sufficient to create an agent which can handle many different situations and which can evolve over time, since excessively numerous rules are needed to answer any input correctly. However, this method is often used as a base combining to other methods to acquire the rule automatically and as result decrease the cost of the rules creation (cf. 1.3.5).

In addition, one of the main drawback of rule-based system is that it is hard to make them real-time and to handle asynchronous input, since the system has to check and evaluate all the rules, in order to finally select one.

1.3.3 Inductive learning

In order to be able to generate a reply to any input a system needs a lot of knowledge. This knowledge can be input in the system before the dialogue starts, however, this solution result a static system which always has the same comportment. In order to make the system evolve and adapt to the user, it can acquire knowledge while acquiring concrete example sentence during the dialogue. The system acquires the knowledge through by the user, and then uses it to improve its future outputs. This process is called *inductive learning* [13]. The inductive learning method can be coupled to a rule-based system in order to increase its flexibility.

In addition, the dialogue agents can share knowledge from several users to be able to output a maximum of input. For example, if one user inputs the sentence “The president of the USA is Obama” and if one other user asks “Who is the president of the USA?” the system can use the first user’s input to answer to the second one’s question. Therefore, if many users use the system a lot of knowledge can be efficiently acquired.

1.3.4 Using the user feedback

In order to adapt to the user and to increase the precision of responses, the dialogue agent can also analysis the user reactions after its output to evaluate it, and then change its comportment in function of the user reactions; i.e in function of the user *feedback*. For example, if the user says “It is not correct”, the system can consider its last output was not correct and do not repeat the same error in future dialogues.

On the other hand, it is also possible to use the user reactions to get the user emotions and to identify what emotion is associated to the system output. For example, if the user inputs “I think it is good”, the system can associate its last response to a *good* emotion.

Especially, the user *feedback* can be considered as part of the *inductive learning* which focuses on the evaluation of the previously acquired knowledge.

1.3.5 Inductive learning method based on genetic algorithm with sexual selection

The *SeGA-ILSD system* [14] uses an *inductive learning* method based on *genetic algorithms with sexual selection* [15] and the user *feedback*. Each sentence is considered as a chromosomes and each word of the sentence as a gene. Concretely, the system acquires rules automatically from pairs consisting of an utterance and its associated response, and attempts to do a crossover of two rules to create a new one. Rules that generate erroneous output are progressively removed from the system using user feedback.

Using chromosomes crossover the system can generate new sentences and as a consequence new rules. One or several words can be exchanged from one sentence to another to create two new sentences. However, in order to avoid non-meaningful sentences it is necessary to only exchange words of the same type such as nouns or verbs. In addition, to increase the robustness of the method it also possible to only exchange words of the same category such as *foods* or *animals*. Figure 1.1 shows an example of crossover where the two words underlined are exchanged.

I ate chocolates $\Rightarrow$ I ate bananas
He likes bananas $\Rightarrow$ He likes chocolates

Figure 1.1: Example of two sentences crossover

Nevertheless, the SeGA-ILSD system uses sexual selection and as consequence, makes a difference between male and female rules. Concretely, females rules contain a simple sentence and a corresponding response. Although male rules contain parameters which can be replaced by any word. For example, the rule “Do you like @1? $\Rightarrow$ I like @1” contains one parameter represented by @1. The system only does crossover of one female rule and one male rule. The male rules ornament them in order to attract female rules. Especially, the male parameters are replaced by concrete words. Then, male and female rules which have common words are used to generate new rules.

In addition, the SeGA-ILSD system also handle mutations. A mutation changes one word of a sentence by another word. Figure 1.2 shows an example of a sentence mutation where the underlined word is changed.

I ate chocolates $\Rightarrow$ I ate bread

Figure 1.2: Example of a sentence mutation

Moreover, in order to increase the rule matching rate, the SeGA-ILSD system uses only the *independent words* of a sentence⁶, as result, two sentences which use different determiners are considered similar, for example “There is a dog” and “There is the dog” are considered similar. This processing allows the system to increase its response rate. Nevertheless, it is highly language-specific and makes the system harder to be adapted to other languages, since language-expert knowledge is required to categorize words.

The SeGA-ILSD system also uses the user feedback in order to execute a selection of rules. Especially, rules which have a *bad* feedback have to be removed of the system, since they not provide a correct response to the user.

In addition, when no rules are found to reply to the input the system uses an ELIZA system to generate the output. The ELIZA system contains manually created rules that are different for each language and as consequence is language-dependent.

1.3.6 Statistical systems

Since many different outputs can be correct for the same input, an efficient system must select the most adequate one. For example, if the input is “How are you?”, “I am fine”, “I am OK” or “I am busy” can be considered as a correct response. As a consequence, the system has to select the best output to show to the user. This process can be carried out using *statistical methods* [16]. Concretely, the system output the most frequently used response. However, this approach must be balanced by other information to select the response in function of the context and the previous outputs to keep the dialogue consistency.

The precision of the statistical method depend on the number of data the system can use. As a result, to be efficient this kind of method requires an important corpus. Nevertheless, it is possible to acquire knowledge from examples [17], in order to collect automatically all the knowledge required by the system.

1.3.7 Main problems

The main problem of all these methods is to handle all the diversity and ambiguity of natural languages. For example, “What is your name?” and “May I have your name?” can be used to ask someone name, however the sentence “May I have my name?” has a totally different meaning; and the sentence “He helps dog bite victim” is ambiguous since

⁶ *Jiritsugo* in Japanese.

the context is required to know if “he” helps the “dog” or the “victim”. In addition, some specific expressions are often used, for example, the expression “break a leg” is composed of the verb “break” and the noun “leg” but means “good luck”; a system looking for each word meaning will not be able to correctly understand this kind of expressions.

Therefore, most of previously-proposed system such as ELIZA, ALICE, SHRDLU or SeGA-ILSD focus on simple sentences and avoid more complicated expressions such as particular idioms. However, the latter are an important part of human speech, and as a result, have to be processed by dialogue agents. In order to solve this problem, flexible methods are required, namely, methods which can look at meaning of word as well as part of word or group of words.

Chapter 2

Language-specific agent

2.1 Introduction

In order to develop a dialogue agent, it is common to limit the possible inputs like did for the SHRDLU system. That is why until now most of the developed dialogue agents focused only on defined languages, even if any input in the target language is allowed, inputs in other languages are prohibited. As a result, these systems can only provide a dialogue in one or several limited languages. Either because of they use language-specific properties such as grammar rules to generate outputs or because they use external tools which are available only in specific languages such as dictionaries or morphological analysis tools. Consequently, they cannot be adapted to any other language easily. This kind of systems are considered *language-dependent*, i.e. *language-specific*.

Moreover, developing a multilingual system requires more works and language expert knowledge to check if the system is really able to handle all the possible cases in all the different languages. Therefore, it is usual to develop *language-specific system* at the first hand. On the other hand, it is possible to hypothesis that most of the user will use standard language which is well-supported by natural language processing tools, and avoid dialect when they speak with a machine. In addition, it is possible to modify a system to make it more language-independent after developed a language-specific version.

Since only a limited number of languages are targeted, language-specific systems can use languages-dependent characteristics. For example, a system targeting an European language can use the spaces present in the sentences to identify words, which is not possible for languages which do not use spaces like Chinese or Japanese. In opposite, the Japanese language subject omission¹ lets system's outputs be correctly interpreted by the user in function of the dialogue context and consequently increases the system performances. However, this kind of tricks cannot be used in all languages. Therefore, some tricks are useful for simple systems, but they can become a trap and make the system harder to be improved. For example, sometime subject has to be explicitly cited even in Japanese to avoid misunderstanding.

¹It is frequent to let the listener inducing the subject of the sentence in function of the context.

Language-specific systems are the most common dialogue agents and can have for example the below objectives. Especially, those objectives are the common objectives of any dialogue agent.

- Carrying out causal dialogues
- Answering questions
- Acquiring knowledge from the user input
- Carrying out dialogues without a long learning phase
- Providing a good output precision

Especially, contrary to language-independent systems, in order to achieve these objectives they can use any external resource or tool. Especially, most of the dialogue agents become language-dependent because they use language-specific tools.

2.2 External resources

Carrying a dialogue requires a huge knowledge, for example, human needs several years of learning to be able to speak a language correctly. That is why most of recent systems use external resources which contain already a lot of information in order to carry out dialogue efficiently without learning phase. For example, dataset for conversational system [18] has been proposed.

The separation between knowledge acquisition and output generation is very useful to develop quickly a system which can handle pleasant and complicated dialogue without requiring an excessively long learning phase. Systems can use simple rules to links inputs to knowledge present in external resources and then output this knowledge to the user. Concretely, as shown in Figure 2.1, external resources are often used to analyze the user input, but they can also be used to select the appropriate output.

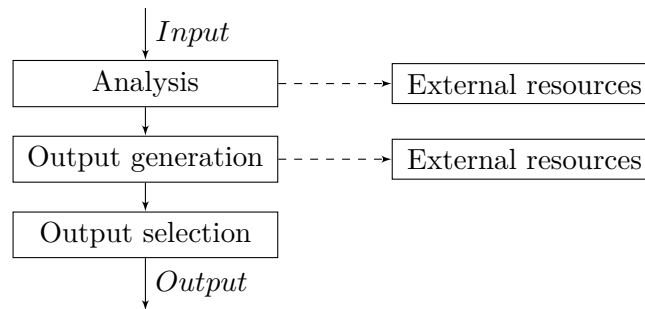


Figure 2.1: Example of external resources utilization

2.2.1 Dictionaries

One of the most easily available resources are dictionaries. Using them, a system can identify words and check their meanings. The meaning, i.e. the definition, of a word is represented using other words. However, using a dictionary to understand the meaning of a word requires to already knowing some basic word meanings.

Figure 2.2 shows an example of English definition², we can see that to understand what is the meaning of “funny”, it is necessary to previously understand other words such as “humorous” or “amusement”.

Causing laughter or amusement; humorous

Figure 2.2: Example of definition for the English word “funny”

Figure 2.3 shows an example of Japanese definition³ for the same word.

満ち足りていて、愉快的気持ちである。
(*Michitarite ite, yukaina kimochidearu.*)

Figure 2.3: Example of definition for the Japanese word “楽しい”

In order to facilitate the word meaning acquisition it is possible to create dictionaries which do not only include words but also information about environment, like images, sounds or gestures, which represent the meaning of the word without using other words. In addition, the structure of the definition of each word can also be adapted for automatic processing (cf. **2.2.3**).

2.2.2 Morphological analysis

In order to proceed to the sentence generation, it is generally necessary to recognize the different words present in a sentence. This task is called *morphological analysis*. It can be quite easily achieved in languages containing explicit markers between words such as spaces or more complicated in case of languages where all the words are juxtaposed like in Chinese or Japanese. That is why for these languages specific tools have been proposed such as JUMAN [19] or Cabocha [20]. Moreover, these tools also provide information about the word’s part of speech⁴ and sometime categorize them in function of their meanings. Basically, these tools use dictionaries of the target language and use it to split a sentence into different words.

Nowadays the precision of *morphological analysis tools* is satisfying, however sometime they failed to split words correctly. For example, the Japanese word “日本人 (Japanese)”

²Extracted from the Oxford Dictionary: www.oxforddictionaries.com/definition/english/funny

³Extracted from the Goo online Dictionary: dictionary.goo.ne.jp/leaf/jn2/138112/m0u/楽しい

⁴verb, adverb, adjective...

is often split in “日本 (Japan)” and “人 (people)”. This kind of unexpected result can have an important influence on dialogue agents and resulting a wrong output generation.

2.2.3 Lexical tools

In order to manage more knowledge and to make it more easily processed by an algorithm many different kinds of knowledge representation have been proposed such as *WordNet* [21], *FrameNet* [22], *OpenCyc* [23] and *ConceptNet* [24]. These lexical databases are structured to be accessed easily and many different APIs (*Application Programming Interface*) exist to use them. Concretely, these resources contain concepts, i.e. words, and relations between them. For instance, they can contain the knowledge: <“cat”, *IsA*, “animal”>.

A simpler method to represent lexical information is to look at words around the target word such as did with the *N-gram model* [25]. For example, the words present around “banana” such as “yellow”, “eat” or “tasty” contain a lot of useful information about the meaning of the target word. The N-gram model can also be used to generate sentences, since it provides words which can be juxtaposed to a target word.

2.2.4 Syntactic analysis

In addition, identifying and understanding the meaning of each word is not enough to understand the meaning of a sentence. It is also necessary to know the relation between each word. For example, in the sentence “I eat ice cream”, “I” is the subject of the verb “eat” and “ice cream” is the target of it. In order to discover these kinds of relation, some *syntactic analysis tools* have been proposed [26], for Japanese *KNP* [27] is often used. However, for example, in the sentence “The sister of my friend with long hair” it is not possible to know which of the “sister” or of the “friend” has “long hair”, it is necessary to check the context of the sentence to eventually be able to choose one.

In addition, *semantic network*⁵ [28] and *frame* [29] can also be used to represent the function of each word and its relations to other words. Therefore, there are various methods to represent knowledge depending on the objectives of the system and the required precision. However, the difficulty is to accumulate enough data and to be able to access it efficiently.

2.3 Rule-based generic system

As explained in the introduction (cf. 2.1), the most common approach to develop a dialogue agent is to use the rule-based method coupled to external tools to analysis the inputs. That is why for the beginning of my research I focused on developing a rule-based dialogue agent targeting only Japanese, and in consequence, which freely uses external tools such as morphological analysis tools and dictionaries. Concretely, I took the *GA-ILSD system* as example, but coded the whole system from zero in order to implement new concepts on a more extensible base.

⁵Also called *frame network*.

The GA-ILSD system has a good learning capability, i.e. it acquires knowledge from the user, however it tends to not reply correctly to questions. Especially, instead of providing a correct response to questions, it simply outputs a sentence or a question containing the same words. This is because the system does not differentiate declarative sentences and questions. Especially, in case of Japanese, questions are frequently differentiated from declarative sentences using a particle which is not considered an independent word, and as a result is ignored by the system when it selects rules.

That is why in order to develop a system which can reply to declarative sentences as well as questions, I considered necessary to differentiate questions and declarative sentences. As a consequence, the system must be able to handle several kinds of rules, those for questions and those for declarative sentences. In order to deal with these different rules, I ordered them in a hierarchy to check them only when a specific kind of sentence is input. As a result, the system does not have to check all the rules it contains and can save time, since it will only check rules corresponding to the input.

In addition, the developed system is not static and acquires continuously rules using the users' inputs, i.e. *inductive learning*, similarly to the original GA-ILSD system. Moreover, since the system is used by several different users, it learns knowledge from different sources, i.e. users, and while crossing this knowledge it creates its own specific knowledge, especially its own tastes and opinions which will be a combination of the different users' inputs. I consider having specific tastes and knowledge is an important point to make the dialogue of the system more natural. Moreover, the system also uses the user feedback to identify and remove incorrect rules.

2.3.1 System architecture

The proposed system uses the same principles as the GA-ILSD system, I also used and improved the *Q-GA-ILSD*⁶ [30] processing of questions to replace interrogative pronouns into a meta-model. Furthermore, the *JUMAN* [19] Japanese morphological analysis tool has been used to obtain the list of the words of a sentence. In addition, the *JUMAN* tool provides meta-data about word's type and also about word's categories. Therefore, I used both of these meta-data to attempt to generate the best response as possible to an input which can be a declarative sentence as well as a question.

The system's data flow is presented in Figure 2.4.

⁶Q-GA-ILSD adds the capability of answering questions to the GA-ILSD system. Basically, it changes interrogative pronouns into a meta-model to match the knowledge that has been acquired and as a result be able to reply to questions.

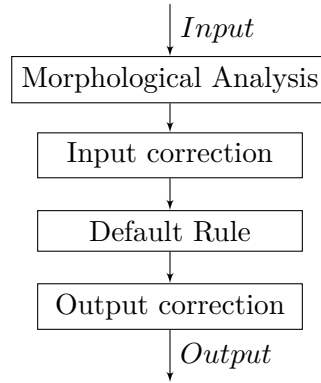


Figure 2.4: System's data flow

First, the *input correction* is basically a personal pronoun replacement, for example “I” is replaced by “you”. If the user says “I am good”, the system will understand “You are good”. The *output correction* can also be used to correct some time relative words, one day after we say “Today is warm”, the output can be changed in “Yesterday was warm”. However, this last correction was not used during the experiment of the system.

Second, the *default rule* is the root rule which contains all the other rules. It takes a sentence as input and checks the hierarchy of the rules to generate an output sentence.

In addition, the whole system is coded in Java, each rule is an object which requires an input stream and generates an output stream. Here, the input stream is the input sentence and the output stream is the response to that sentence. All rules created, all classes coded, can be considered as a *dialogue agent framework*, since they can be inherited by other classes to change the system behavior. In addition, the system does not use external database such as MySQL⁷, in order to access the rules as fast as possible.

Main concepts

In this section, I present the main concepts used by the system. They were used to develop the structure of the dialogue agent.

Learning At the beginning, the system has no specific knowledge and is expected to evolve differently in function of the situation. I considered if too many static knowledge is provided to the system this will influence its evolution. Therefore, the system has to prioritize acquiring knowledge from the user's input, that is called *learning*. Concretely, the system saves all the users' inputs and acquires knowledge they contain. For example, the system saves the inputs and the relations⁸ between sentences and between words.

Statistics While speaking and interacting with the user, the system acquires many knowledge, sometime the same knowledge is acquired several times. The more frequently

⁷<https://dev.mysql.com>

⁸question/answer, sentence/response, etc...

a knowledge is learned, more important it becomes. For example, if the system acquires the input “I am a boy” two times and the input “I am a girl” one time, and then, if the user asks “What are you?” the system will output “I am a boy”.

Evolution In a similar way as the GA-ILSD algorithm, the system uses genetic algorithm to evolve and to delete wrong rules. In order to know if a rule is wrong or not, after using it, the system analyzes the user’s reaction, i.e. the user new input, in order to obtain the user *feedback*. If this reaction contains specific keywords⁹, the rule previously used is considered *incorrect*. In other cases, the rule is considered *correct*. Then, the rule *adaptation rate* (cf. **2.3.1**) is updated in function of the evaluation: *incorrect* or *correct*.

Abstraction When humans speak to each other, sentences can be interpreted on many different levels. As we can see in Figure 2.5, a same sentence can be processed to generate several other sentences related to it. For example, data can be added from the context or removed in the sentence, such as only keeping some words or only using meta-data such as word part of speech or word’s category to create a *meta-model*. I consider more the sentence contains precise data more the level is concrete, in an opposite manner, less data are precise more the level is abstract.

Input:	今日、晴れです。 (Today the weather is clear.)
Adding context’s location:	今日、札幌で晴れです。 (Today the weather is clear in Sapporo.)
Removing temporal data:	晴れです。 (The weather is clear.)
Taking part-of-speech:	< 動詞 > ^a (<verb>)
^a Extracted from the verb “晴れる (to be sunny).”	

Figure 2.5: Example of sentence levels

Meta-model In order to increase the abstraction level, the system also uses *meta-model* which represents a set of word such as all the nouns or all the words which can be used to answer to a *wh-question*. For example the model “<nouns>” matches all the common nouns, while the model “<sport>” matches all the words which are in the “sport” category¹⁰.

⁹The same keywords list as the GA-ILSD system is used.

¹⁰Categories can be provided by the JUMAN morphological analysis tool.

Rules

The proposed system uses rules to generate output and is, as result, *rule-based*. Each rule can be called with an input sentence and it generates an output sentence if it is possible. However, in order to generate the most suitable output it is necessary to check many different rules and to select the best response to show to the user. That is why those rules are hierarchically organized and are classified in four categories that are explained in the next sections. We can see an example of a hierarchic structure of the rules in Figure 2.6.

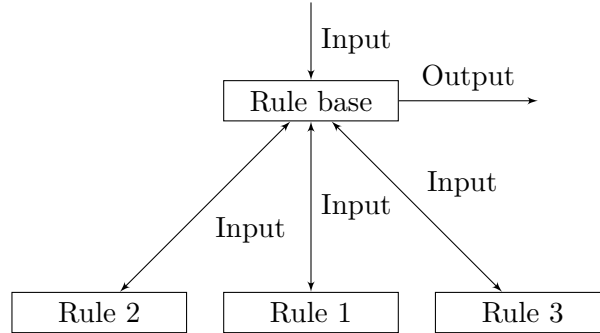


Figure 2.6: Example of a structure of the rules

New acquired knowledge are saved as a new rule¹¹ which is referenced in a *rule base*. When a user inputs a question, the *rule base* checks all the rules it contains to find a rule which can generate a response.

Simple rule A *simple rule* is very simple as its name suggests. We can see its data flow in Figure 2.7. It contains two sentences; a *matching sentence* and an *output sentence*. If the user's input sentence which matches the rule's matching sentence, the rule will reply its output sentence. In other cases, the rule replies nothing.

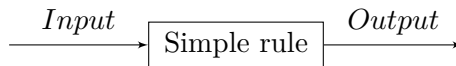


Figure 2.7: Simple rule data flow

In Figure 2.8, “How are you?” is the matching sentence and “I am fine” is the output sentence. The rule can reply only when the input sentence is “How are you?”.

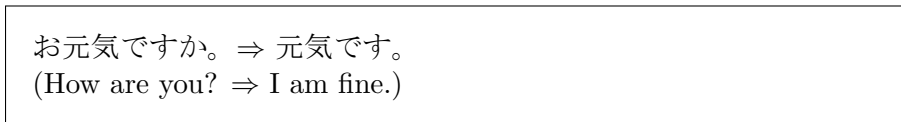


Figure 2.8: Example of a simple rule

¹¹For example: Rule 1, Rule 2, Rule 3.

Rule base A *rule base* is a rule which contains other rules. Especially, it is simply a list of rules. As we can see in Figure 2.9, when the rule is asked to generate an output, it will look for the most suitable rule to reply among all the rules it contains. If no rule matches the input, the rule does not provide any response.

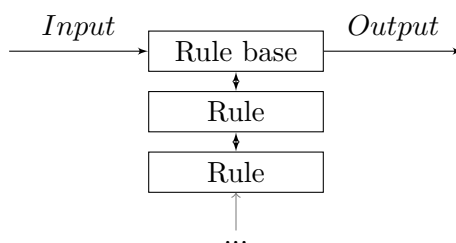


Figure 2.9: Rule-based system's data flow

Default rule The system's first rule which contains all other rules is called the *default rule*. It could be also called the *root rule*.

Abstract rule An *abstract rule* contains a reference to one or several other rules. As shown in Figure 2.10, when the rule attempts to generate an output, it looks for the knowledge contained in referenced rules to generate its own response. Unlike the *rule base* which directly uses other rule's output, the abstract rule amends the data using a specific strategy to generate a new response. Moreover, it can use the data contained in several rules to generate one output. For example, an *abstract rule* can combine two rules to generate a response. If the input is “What do you like?”, the rule will search *simple rules* which match the input, find “I like cats” and “I like dogs”, and generate the output “I like cats and dogs”.

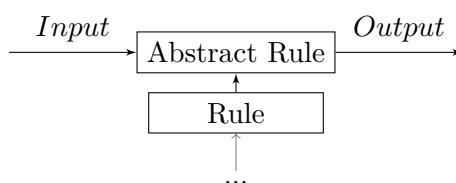


Figure 2.10: Abstract rule data flow

Recursive rule A *recursive rule* changes the input sentence before sending it to another rule (cf. Figure 2.11). *Recursive rule* adds or removes a part of the input sentence to ask another rule to reply to this new input. For example, if the user says “今日は晴れですね (Today is clear weather, isn't it?)”, a recursive rule can remove not meaningful parts of the sentence¹² to create a new input “今日晴れ (Today clear weather)”.

¹² *Fuzokugo* in Japanese.

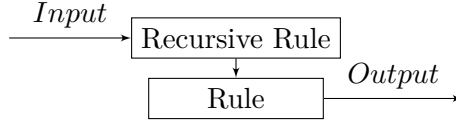


Figure 2.11: Recursive rule data flow

One of the best example of *recursive rule* (cf. Figure 2.12) is the rule which changes interrogative pronouns into a meta-model. For example, “who” is changed into the meta-model which matches all the human names. Then, this new sentence is sent to the *default rule* to attempts to reply to the question using the new input.

友達は誰ですか。⇒ 友達は < 人名 > です。
 (Who is your friend?⇒<name> is your friend)

Figure 2.12: Example of a recursive rule output

Adaptation rate

In order to choose the best rule to reply to the user among all the possible ones, the system calculates the *adaptation rate* of each rule, and then the rule which has the highest adaptation rate is chosen. We can see the formula used on Equation (2.1). This formula is inspired by the GA-ILSD’s adaptation rate. However, I added new variables in order to make the same formula able to handle any kinds of rules and to be able to hierarchy them.

Simple rules have an *importance coefficient* k , superior or equal to one and a *reduction coefficient* g , equal to one. However, *recursive rules* and *rule base* have an *importance coefficient* of one and a *reduction coefficient* different to one. This is because these rules do not generate an output, but use other rules which already have an adaptation rate, and as a consequence the systems only needs to change it.

$$t_a = g \times k \times \left(\frac{nb_c}{nb_c + nb_i} + h \right) \quad (2.1)$$

t_a Adaptation rate

g Reduction coefficient

k Importance coefficient

nb_c Number of correct use

nb_i Number of incorrect use

h Number of acquisition

To generate a response the system often uses several different rules. Therefore, the *adaptation rate* changes while combining different rules. The system uses Equation (2.2) to calculate the final rule's *adaptation rate*.

$$t_a = \prod_{i=1}^{n-1} t_{a_i} \quad (2.2)$$

t_a Final adaptation rate

n Number of used rules

t_{a_i} Adaptation rate of the i used rule

Setting of the system

The system has been created to be as flexible as possible, as a consequence, many different settings are possible. Nevertheless, I chose to focus on opinions and tastes acquiring and to disable other capabilities¹³. Since I first considered making the system as reactive as possible and reduces the bug's number.

Used rules

As explained in **2.3.1**, the system uses many different kinds of rule. Moreover, there are many different instances of each of them. Concretely, about 30 different classes of rules have been created. The most important of those rules, those used in the experiment, are listed below.

- *DefaultRule* is the first rule the system uses to reply an input sentence. The *default-Rule* is a kind of *rule base*.
- *RepeatingRuleBase* is a *rule base* which contains *simple rules* replying using the input sentence as output. When the user inputs a new sentence, the rule generates a new simple rule to save it.

For example, if the user inputs the sentence "Today, I go to school" a simple rule which replies "Today, I go to school" is generated.

- *DeclarationRuleBase* is a *rule base* containing *simple rules* which reply to *declarative sentence* using a sentence learned from the user. When the user inputs a new sentence, the rule generates a new simple rule to save the relation between the previous sentence and the new sentence.

For example, if the system says "I am fine" and the user replies "Happy to heard that", a simple rule which replies "Happy to heard that" when the input is "I am fine" is generated.

¹³task rule, sentence generation rule, etc...

- *QuestionRuleBase* is a *rule base* containing *simple rules* which reply to *question* using a sentence learned from the user.

For example, if the system says “How are you?” and the user replies “I am fine”, a simple rule which replies “I am fine” when the input is “How are you?” is generated.

- *ApologyRule* generates an apology sentence such as “Sorry” if the input contains a keyword like the sentence “You are wrong” which contains the keyword “wrong”.
- *JiritsugoOnlyRule* is a *recursive rule* which removes *not meaningful words* (*Fuzokugo*) from the input sentence.
- *InterrogativePronounRule* is a *recursive rule* which replaces interrogative pronouns by a *meta-model*.

All the rules excepting the *DefaultRule* are encapsulated in a *rule base*. For example, as shown in Figure 2.13, the *InterrogativePronounRule* is encapsulated in the *DefaultRule* and sends amended input to the *DefaultRule*. *InterrogativePronounRule* is used only one time since amended data contains no pronouns anymore, only pronouns’ meta-model.

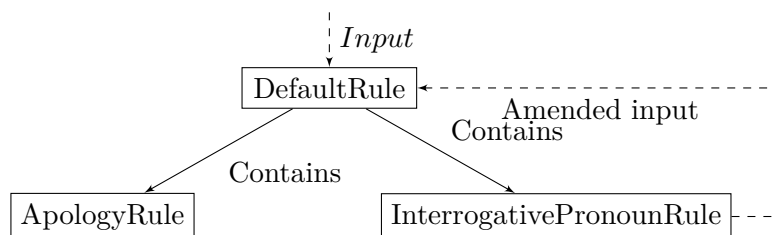


Figure 2.13: Example of rules arrangement

Characteristics of the rules Table 2.1 shows the coefficients set to each kind of rule. Rules which reply to questions, such as *QuestionRuleBase* and *InterrogativePronounRule*, have a higher coefficient than other in order to reply question correctly and to avoid selecting a rule supposed to be used for declarative sentence.

Table 2.1: Rules characteristics

Rules	Coefficient	Reduction coefficient
SimpleRule	1	1
DefaultRule	1	0.9
RepeatingRuleBase	1	0.9
ApologyRule	1	1
DeclarationRuleBase	1	1.2
QuestionRuleBase	1	1.4
JiritsugoOnlyRule	1	0.9
InterrogativePronounRule	1	1.6

For example, if the input sentence is “お元気ですか？(How are you?)” and a *simple rule* contained in a *QuestionRuleBase* can reply to it. The final *adaptation rate* will be $0.9 \times 1.4 \times 1 = 1.26$ (with $nb_c = 0^{14}$ and $h = 1$). If the input sentence is “猫は何ですか？(What are cats?)” and a *simple rule* contained in a *RepeatingRuleBase* can reply to “猫は動物です (Cats are animal)”. The system can access this knowledge through an *InterrogativePronounRule* which will generate the new input “猫は < 名詞 > です (Cats are <nouns>)”, and the final *adaptation rate* will be $0.9 \times 1.6 \times 0.9 \times 0.9 \times 1 = 1.1664$ (with $nb_c = 0$ and $h = 1$).

Number of rules

The starting system has 48 rules, however, for example, if “Hello” is input the number of rules increases to 53. At the end of the experiment the system contained about 2000 rules. Counting knowledge is very hard, however, each rule is considered as an atomic knowledge.

All the new inputs are saved in one or several simple rules. That is why the number of rules increases more than one after each new input.

2.3.2 Behaviour

In the similar way as a human, I consider a dialogue agent has its own *behavior*. Here, *behaviour* means how the system replies to some kinds of inputs, like questions or declarative sentences.

Aiduchi

If the system cannot give a proper response using rules, it uses *Aiduchi* [31] to provide a response. *Aiduchi* are interjections showing that we understand and are listening what the speaker is saying. We can see in Figure 2.14 an example of *Aiduchi*.

¹⁴First use of the rule.

猫は可愛いです。⇒ そうですね。
(Cats are cute. ⇒ That is right.)

Figure 2.14: Example of *Aiduchi*

Declarative sentence

When the user inputs a declarative sentence, the system attempts to output a question which matches the input, and then if there is no one, it looks for a declarative sentence. Finally, if there is no knowledge about the user's declaration the system answers with an *Aiduchi* [32]. Figure 2.15 shows some of these cases. It is also possible to add similar capability as the Teruhisa's Guidance system [33] to give much more information to the user when it is possible.

晴れです。⇒ 晴れが好きですか。
(The weather is clear. ⇒ Do you like clear weather?)
猫が好きです。⇒ 猫は可愛いですよ。
(I like cats. ⇒ Cats are cute.)
学校に行きます。⇒ そうですね。
(I go to school. ⇒ Really?)

Figure 2.15: Response of a declaration

Question

If the input is a question, the system attempts to answer it, for example through the *InterrogativePronounRule* which will change interrogative pronouns in a meta-model which can match *simple rules*. If it cannot, it repeats the question to the user. I considered this is a little unnatural behaviour. However, one of the objectives of the system is to acquire the user's feelings and opinions, therefore it has to ask the user to learn how to answer to the question. Therefore, as shown in the example of conversation of Figure 2.16, the system replies "How are you?" to the question "How are you?", but I think this response is in fact similar to "I do not know, how about you?".

お元気ですか。⇒ お元気ですか。
(How are you? ⇒ How are you?)

Figure 2.16: Response to a question

2.3.3 Knowledge's sharing

The system uses two methods to acquire knowledge. Some basic knowledge¹⁵ is pre-tough to the system with an AIML file, others are acquired during the conversation with users. Concretely, I used a very simplified AIML representation to save pair of sentence.

Since not only one user uses the system, one user's knowledge can be used to reply to another user's input sentence. That is what is called *knowledge's sharing*. While acquiring many data from many different users the system can generate its own behaviour and acquire its own tastes. I studied this capability more in detail in 2.4.

2.3.4 Saving

The state of the system is saved after a conversation finish and reloaded when another user starts a conversation. If two users use the system at the same time, the data of the user who finishes later erases previously saved states. However, all the sentences are saved on a back end rule which is never deleted and used to generate response when system cannot do it with its own knowledge. Consequently, knowledge is never lost.

2.3.5 Interfaces

In order to realize the experiment a self-made *Android application* is used (see screen-shot in Figure 2.17). The system is also accessible on a *web page*¹⁶. I also created an *iOS application* (see screen-shot in Figure 2.18) and a web interface (see screen-shot in Figure 2.19), but they were not used for experiment.

¹⁵Basically system's information like name or age.

¹⁶<http://demo.media.eng.hokudai.ac.jp>



Figure 2.17: System's interface on Android

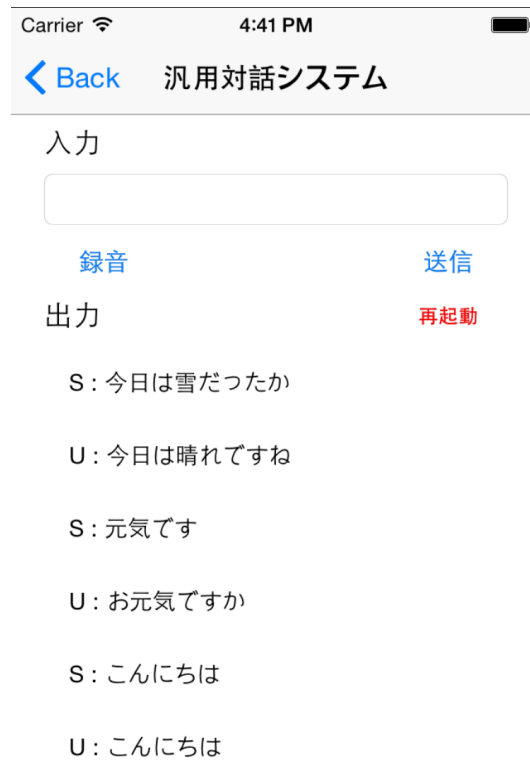


Figure 2.18: System's interface on iOS

自然言語処理を用いたシステム		
ELIZA	言語獲得能力付き対話システム	質問獲得能力付き対話システム
汎用対話システム	汎用対話システム保存なし	知識獲得機能付き対話システム(単一ユーザ)
知識獲得機能付き対話システム(SAMU)	好み獲得機能付き対話システム (SDPATK)	ユーザ別知識獲得機能付き対話システム (SAEU)
倫理判断システム	駄洒落生成システム	新駄洒落生成システム
漫才生成システム	新漫才生成システム	雑談システム (ベースライン)
知識獲得システム (ベースライン)	感情認識システム	感情認識機能付き対話システム

注) ここで使用できる実験システムは、デモのために作成されたもので、使用するデータや処理時間等の制約から本来の性能ではありませんが、どのようなシステムであるかを知っていただくために公開しているものです。

ウェブソケットに接続しました。データを送れます。

利用しているシステムは「汎用対話システム保存なし」です。

ユーザ

以下入力して下さい。

対話

S: 元気です

U: お元気ですか

S: こんにちは

U: こんにちは

Figure 2.19: Web interface of the system

Voice input is available on the applications, but I asked experiment subjects to only use keyboard to input their sentences in order to avoid speech recognition errors.

Using the application on a cell phone permit me to realize experiments every time and everywhere using a 3G¹⁷ connection to connect to the server which hosted the system.

In addition, I also implemented the system in a robot: *RAPIRO*¹⁸ (see Figure 2.20). This interface uses voice recognition result as input and provides an response to the user using speech synthesis.

¹⁷Third generation of mobile telecommunications technology

¹⁸<http://www.rapiro.com/>



Figure 2.20: Robot used as interface of the system

2.3.6 Experiments

In order to prove effectiveness of the propose system and the importance of having opinions and tastes to chat with humans, I considered to compare the developed system with an ELIZA based system¹⁹.

The experiment uses the semantic differential approach [34, 35] with the same procedure as Uchida's paper [36]. I asked subjects to answer some questions, to test the two systems, and I asked subjects' impressions after they tried each one. First system is randomly chosen and only some basic information about the system and how to use it is provided. Moreover, I asked subjects to input about thirties grammatically correct sentences in each system. After using one system the subject fills out the survey's form corresponding.

Three below introduction questions has been asked:

¹⁹Accessible at: <http://demo.media.eng.hokudai.ac.jp>

- Did you know spoken agent?
- Did you already use a spoken agent? If you did, which one?
- Do you prefer typing input sentence or use voice recognition?

At the end of the experiment, I concluded by asking what part of the dialogue was the most impressive. I used these question's answers to obtain a more subjective impression's result about each system.

Subjects

Table 2.2 shows information about subjects who participates in the experiment. I looked for result's differences between each category of subject, but no one really appeared, results do not vary in function of these parameters.

Table 2.2: Subject's information

	Number
Subject	14
Male	8
Female	6
Student	6
Worker	8
Average age [year]	24.64

Experiments results

As shown on Table 2.3, only half of the subjects knew what was a dialogue agent before the experiment. However, I think that many subjects already used a dialogue agent, but did not realize it. For those who already used one, the name of Siri²⁰ and i コンシェル²¹ (*iKonsheru*) was the most frequent.

Table 2.3: Results of the preliminary questionnaire

Question	Yes [%]
Know spoken agent	50
Used spoken agent	36
Prefer voice input	50

²⁰<https://www.apple.com/jp/ios/siri/>

²¹<http://www.nttdocomo.co.jp/service/customize/iconcier/>

Figure 2.21 shows a successful dialogue’s part which occurred several times during the experiment.

お元気ですか。⇒ 元気です。
 (How are you? ⇒ I am fine)
 好きな食べ物は何ですか? ⇒ 味噌汁です。
 (What food do you like? ⇒ It is miso soup.)

Figure 2.21: Experiment dialogue example

Table 2.4 and Figure 2.22 show the results of the semantic differential form.

Table 2.4: Semantic differential experiments results

Adjective		System	ELIZA
Bad	Good	4.86	4.29
Odious	Cute	4.36	3.29
Boring	Interesting	4.50	4.57
Malicious	Kind	4.86	3.42
Idiot	Clever	4.57	4.14
Inconvenient	Useful	3.79	3.29
Won’t use	Want to use	4.14	4.29
Empty	Full	3.93	3.50
Tedium	Have many interest	5.43	5.43
Complicated	Simple	3.64	4.71
Machine-Like	Human-Like	3.43	3.36
Slow	Fast	4.57	5.86
Hard to be intimate	Easy to be intimate	4.14	4.07
Hard to understand	Easy to understand	4.86	4.29
Disappointed	Satisfied	4.86	4.36

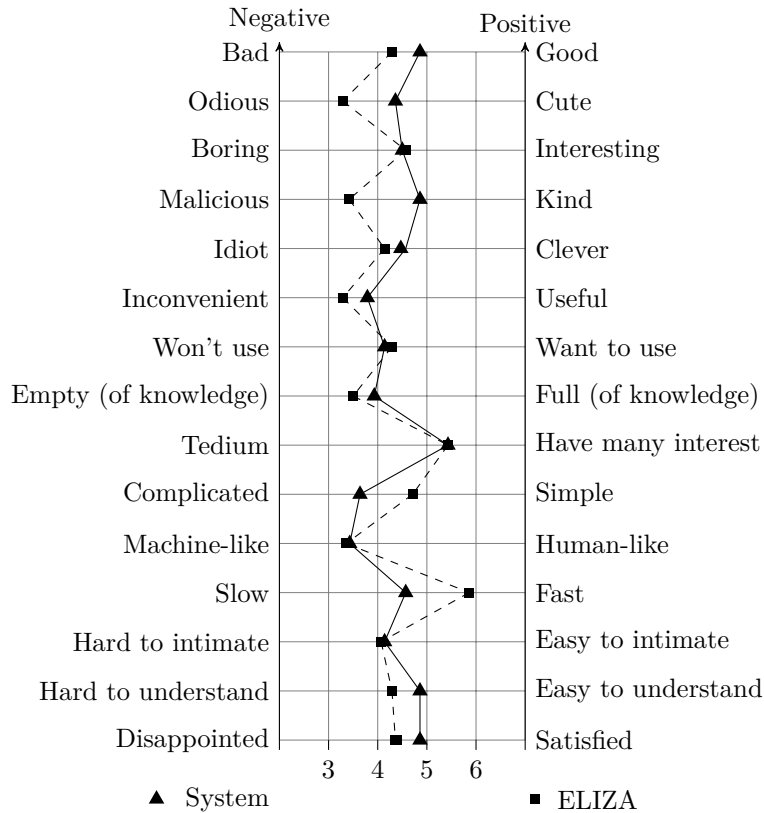


Figure 2.22: Semantic differential experiment's results

Results analysis

In general, the new system is a little bit better than a simple ELIZA system, but results are often very similar, many subjects prefer the ELIZA system on some points. The new system's mean result is 4.40 and ELIZA's mean result is 4.19, but ELIZA obtained better results on two points; *interesting* and *fast*. Moreover, five subjects give a better mean result to the ELIZA system.

The new system was still very basic and some subject's speaking such as dialect was not correctly handled by the system. Even if users were asked to use grammatically correct sentences and standard Japanese, some inputs which were supposed to be correctly answered just obtain an *Aiduchi* as response, because no rule matches the user's input sentence due to some words or grammatical differences which do not change the meaning of the sentence.

If we look in details, we can see that proposed system is especially more *cute*, *clever* and more *full*, but not really *better* or *wanted to be used* more. I think users feel that the system was more powerful, but during the conversation the system replies to questions correctly and naturally only a few times. Consequently, I think the system knowledge was not sufficient to obtain good results, but potentially it can obtain better results if more

knowledge is taught to it. Moreover, even if the system contains the adequate knowledge if a question is constructed differently the system will not be able to match the correct rule. For example, “What food do you like?” and “What are your liked foods?” have the same meaning, but different constructions, and as result, the system does not understand it as one single similar question.

In particular, subjects who used non-standard Japanese encounter some problems with both systems (proposed and baseline system) especially due to morphological analysis errors. However, the proposed system encounters more difficulties, since if a user uses a non-usual sentence the system will not match the knowledge it acquired until there and in an opposite manner, if the user teaches non-standard inputs, other users will not be able to access this knowledge too.

The ELIZA’s system responses are very fast and generation time does not change a lot in function of the input length. However, the proposed system attempts to find the most adequate response to the user’s input, and as a result question’s answering can become longer and increase in function of the length of the input and of the quantity of previously acquired knowledge.

In addition, even every subject’s dialogues and impressions was quite different, every subject enjoyed obtaining a proper response to some questions. However, the system uses too many *Aiduchis* and was quickly annoying after some turns.

2.3.7 Conclusion

I developed a new dialogue agent which learns tastes and opinions from users to be able to reply many kinds of question. During the experiment, the systems provided some nice dialogues where the user could enjoy speaking with a dialogue agent which looks natural and has its own personality. However, the system was not perfect and needed to be enhanced. The simplest way to increase the system response quality would be to teach it more knowledge, nevertheless this would make the system too slow to be used. That is why I thought a better strategy has to be developed to find the best response in the fastest way and to not have to check all the system’s database at any new input. I studied some possible solutions in **Chapter 3**.

Nevertheless, I could show that being able to reply some simple questions let the users have better and more impressive conversations than a simple ELIZA system which replies to question with other questions. Most of subjects understand the new capacities of the system and tried to teach it some knowledge, but some subjects only asked questions without replies to the system’s one. Consequently, as a first improvement I thought about adding other ways to acquire knowledge such as using social networking service and other ways to provide this knowledge to the user. In the next section, I studied several methods to share knowledge between several users.

2.4 Knowledge sharing

The fundamental point of a dialogue agent is the knowledge acquisition, more knowledge has a system more input it can reply. However, the used knowledge has to be adapted to the user. For example, a system with many knowledge about cooking will not be suited for someone who wants to have a dialogue about sport. That is why I considered that the knowledge has to be acquired from the user himself or from resources related to the user. Therefore, instead of use external corpora I chose to acquire knowledge from all the users of the system (they are related to each other since they all use the same system) and to share knowledge learned to provide it to another.

In addition, since there are many ways to use a dialogue agent, it has to be adapted to the situation where it is used [37]. I think that all the users do not expect to speak to a machine like to a human, but expect their wishes to be suited. These wishes can be very various and I consider some users prefer a machine to be more precise and strict than a human. Consequently, speaking in a human way is maybe not as important as generally thinking.

That is why I studied the best method for a machine to reply to the user's questions. I imagined two new methods to answer questions and implemented them in the previous presented language-specific dialogue agent. I created two new systems, one which acquires knowledge, makes it own to reply to the questions and one which acquires knowledge and transfers it directly to reply to the user's questions.

The research objectives were to compare these two kinds of question answering and to study the user's impressions about them. I think those two kinds of answering have their own utilities, but they can be more useful in a specific situation. For example, while using at home or at work, one of the systems is maybe more suitable.

Next, I explain both systems. Then, I compare the two system's answers helped with carried out experiment. Finally, I analyzed the differences of these two kinds of question answering.

2.4.1 System architecture

I used the same base as the previously presented system to create two new systems. Consequently, the previous system is used as a framework which already acquires knowledge from users automatically. Nevertheless, I had to change how the system generates responses using this knowledge to change its comportment. Concretely, I changed rules which are used to generate the responses to the questions.

Figure 2.23 shows the input's processing. All the users' inputs are transferred to *shared rules* which can be used by any user's rule. Consequently, each user's knowledge is gathering up, and then can be used to generate a response.

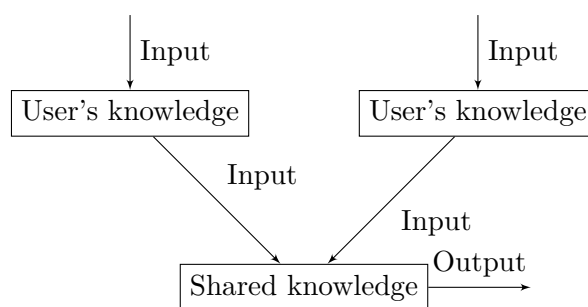


Figure 2.23: Processing of user's input

2.4.2 Two different methods to share knowledge

To realize this part of the research I modified the previously explained system to create two new dialogue agents which have two different behaviors.

Knowledge acquisition function for multi-user

Spoken Dialogue System with Acquiring Knowledge Function for Multi-User^{22,23} has been developed to be useful when it is used by several users. It acquires knowledge from all the users, and then uses the most often acquired knowledge to generate the response.

Figure 2.24 shows an example of dialogue for SAMU.

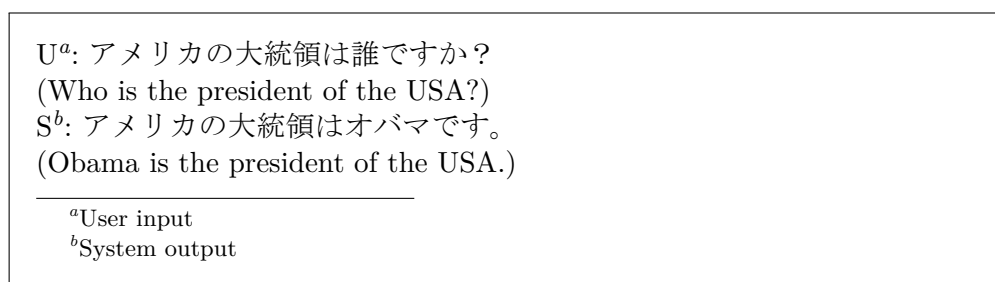


Figure 2.24: Example of a dialogue for SAMU

In Figure 2.24, we can feel that the system's response is quite *human-like*. The response is syntactically and grammatically correct, moreover this response has been learnt from a human who replied to the same question like that. In the response, we do not know who taught that knowledge to the system, but the system reply like if it used its own knowledge. The system chooses to reply "Obama is the president of the USA." since until now most of the people taught it this knowledge. However, maybe, some people said "Bush is the president of the USA."

²²Shorted by SAMU.

²³Available on : <http://demo.media.eng.hokudai.ac.jp>

I think that behavior is the same as a human who chooses in general the knowledge he heard the most often until now to reply to a question. However, if the president of the USA changes, the system will continue to reply the name of the previous president until more people teach it the new name of the president or until someone tells the system it is wrong. In that case, using feedback the system can avoid wrong response quickly.

Knowledge acquisition function for each user

Spoken Dialogue System with **A**cquiring Knowledge Function for **E**ach **U**ser^{24,2} has to be used by several users too. It acquires knowledge from each user and uses all the knowledge it learned from all the users to generate the response.

Figure 2.25 shows an example of dialogue for *SAEU*.

U: アメリカの大統領は誰ですか?
(Who is the president of the USA?)
S: アメリカの大統領はオバマですと花子さんと慶太さんが言いました。
アメリカの大統領はブッシュですと卓郎さんが言いました。
(Hanako and Keita said that Obama is the president of the USA. Takuro
said that Bush is the president of the USA.)

Figure 2.25: Example of a dialogue for *SAEU*

Hanako, Takuro and Keita spoke with the system, all of them taught a knowledge which can be used to reply to “Who is the president of the USA?”. Hanako and Keita taught the same knowledge to the system, “Obama is the president of the USA.” and Takuro taught “Bush is the president of the USA.”. The system does not look for the best response, but simply gives all the knowledge to the user. Each user’s input is considered as a different knowledge even if several users said exactly the same sentence.

I think this behavior is a little more *machine-like*. In general, the human attempts to avoid useless or wrong parts of speech. If we ask someone “Who is the president of the USA?”, he will certainly only reply the name of the current president and not cites from whom he learned that name. That is why I consider this system is not a simple dialogue agent, but a knowledge sharing system using natural language.

When someone works in a group it is hard to transfer information to other people in a smooth way. Therefore, I think this kind of system can be a funnier way to exchange knowledge. For example in a group of few people, if we want to know if someone like curry we can ask the system “Do you like curry?” and the system will teach us what thinks each member of the group.

In addition, I considered many users do not expect to chat with a machine like with a human, but they just expect to receive the information they want. Consequently, even

²⁴Shorted by *SAEU*.

if a system is more machine-like, I think it can be more useful than a human-like system. Most of the machines are seen as a tool to obtain objective results. If we have a simple question it is very easy to ask someone who is near you to obtain a response. However, in this case the response is subjective and may be wrong. SAEU's response is as complete as possible and lets the user choose which knowledge he believes or ignores.

2.4.3 Experiments

The experiment focuses on the evaluation of both systems' question answering. I did not study other parts of speech such as declarative sentence or greetings. I also compared the results to two baselines, ELIZA system's responses and human's responses.

Preparation of the experiments

In order to obtain a set of responses generated by both systems I asked subjects to chat with both systems. First, they gave some knowledge to the systems and then they input questions in the systems.

12 people participate in this first phase. I asked them to teach simple knowledge and to ask simple questions to the systems about a specific topic in order to increase the systems' answering rate. As result, I obtained a set of question/response for both systems.

Table 2.5 shows information about people who participates to this work.

Table 2.5: Preparation phase subject's information

	Number
Subject	12
Male	7
Female	5
Student	8
Worker	4
Average age [year]	24.64

Each participant taught about 30 knowledge (sentences) to the system, therefore at the end of this phase the system collected about 360 sentences containing knowledge. However, each sentence is used to generate several rules, consequently at the end of the experiment the system contained more than 1,000 rules.

In order to generate a set of question/response for the first baseline, 20 questions have been selected randomly from the first phase dialogues, and then, responses have been generated using an ELIZA system²⁵.

²⁵Available on: <http://demo.media.eng.hokudai.ac.jp>

A set of human answers has also be generated. I asked five people to response to 20 questions chosen randomly in the same dialogue corpus. Then, I asked them to answer using the same specific topics used during the preparatory phase.

Since humans adapt their way of speaking when they speak to a machine [38], I wanted to only use machine oriented questions. That is why I used the same set of questions to create the baseline and the human answers.

We can see examples of samples for SAMU, SAEU, ELIZA and for humans in Figure 2.26, 2.27, 2.28 and 2.29.

U: 好きな食べ物では何ですか？
(What food do you like?)
S: 私はシュークリームが好きです。
(I like cream puff.)

Figure 2.26: A dialogue sample for SAMU

U: 何を食べますか？
(What do you eat?)
S: 丼を食べました。と広樹さんが言いました。
(Hiroki said he ate Donburi.)

Figure 2.27: A dialogue sample for SAEU

U: 食事をしますか？
(Do you have meal?)
S: 食事ですか？
(Meal?)

Figure 2.28: A dialogue sample for ELIZA

U: カレーを作るのは好きですか?
(Do you like cook curry?)
S: 普通かな。
(So-so.)

Figure 2.29: A dialogue sample for a human

Experiment process

I used a semantic differential approach [35, 34] to compare the two new systems' responses and the baselines. The same set of adjectives as the previous experiment is used in order to facilitate comparisons. Moreover, I also asked several questions to know, for example, if the answers were natural or not and which system the subject would choose to use.

Experiment's steps

The subjects had to evaluate 20 answers randomly chosen from the set of samples on each system, (5 questions/responses for each system). I asked them to evaluate the response as a system response and not as a human one. After filling out a form, a new sample is displayed. The list of adjectives contained in the form is randomly sorted and the subject's previous answers are cleared.

I also asked the subjects to choose if the answer is *human-like* or *not*, if the answer *has meaning* or *is unnatural* or *is natural*, and if the answer *contains knowledge* or *not*.

At the end of the experiment I explained the three systems' behaviors more in details. Then, I asked the subjects the three questions below.

- Which system do you want to use?
- Which system do you want to buy?
- Which system is useful?

Experiment interface

In order to be sure all the samples and all the forms are randomly organized I created a simple interface using the *Swing Java GUI widget toolkit* [39]. The subjects used it to fill out the experiment form.

Figure 2.30 shows a screen-shot of the interface used for the experiment.

質問	昨日は何を食べましたか？							
応答	どんぶりを食べました。							

	非常に	かなり	やや	どちらもない	やや	かなり	非常に	
便利な			0					不便な
可愛い				0				にくらしい
邪魔する					0			使いたい
満足した				0				がっかりした
愚かな			0	0				賢い
機械的な					0			人間的な
速い		0				0		遅い
親しみにくい		0						親しみやすい
分かりやすい			0					分かりにくい
意地悪な					0			優しい
面白い				0				つまらない
充実した			0		0			空虚な
単純な						0		複雑な
退屈な				0				興味深い
良い		0						悪い

評価

18

Figure 2.30: Experiment interface

Subjects

In this second phase of the experiment, subjects who did not participate in the first part evaluated question/response samples of each system. Table 2.6 shows information about this second group of subjects. I looked for result's differences between the subject categories, but no one appeared.

Table 2.6: Subject's information

	Number
Subject	10
Male	6
Female	4
Student	7
Worker	3
Average age [year]	25.50

Experiments results

Table 2.7 shows the result of the questions asked to each subject.

Table 2.7: Experiment’s result

Adjective		SAMU	SAEU	ELIZA	Human
Bad	Good	4.55	4.16	2.80	5.39
Odious	Cute	4.16	4.08	3.34	4.60
Boring	Interesting	4.54	4.14	3.40	4.78
Malicious	Kind	3.94	4.21	2.86	4.52
Idiot	Clever	4.60	3.96	2.82	5.02
Inconvenient	Useful	4.25	4.00	2.94	4.80
Won’t use	Want to use	4.24	3.62	2.79	4.78
Empty (of knowledge)	Full (of knowledge)	4.42	3.94	2.84	4.57
Tedious	Have many interest	4.84	4.26	3.32	4.66
Complicated	Simple	4.56	4.10	3.54	4.84
Machine-like	Human-like	4.74	2.94	3.52	5.44
Slow	Fast	4.00	3.80	4.24	4.12
Hard to be intimate	Easy to be intimate	4.04	3.70	2.94	4.99
Hard to understand	Easy to understand	4.50	3.92	2.78	5.18
Disappointing	Satisfying	4.84	4.12	2.90	4.92
Mean		4.41	3.93	3.13	4.84
Mean (Machine-like/Human-like excluded)		4.34	4.00	3.15	4.76

Figure 2.31 shows the result of the semantic differential experiment.

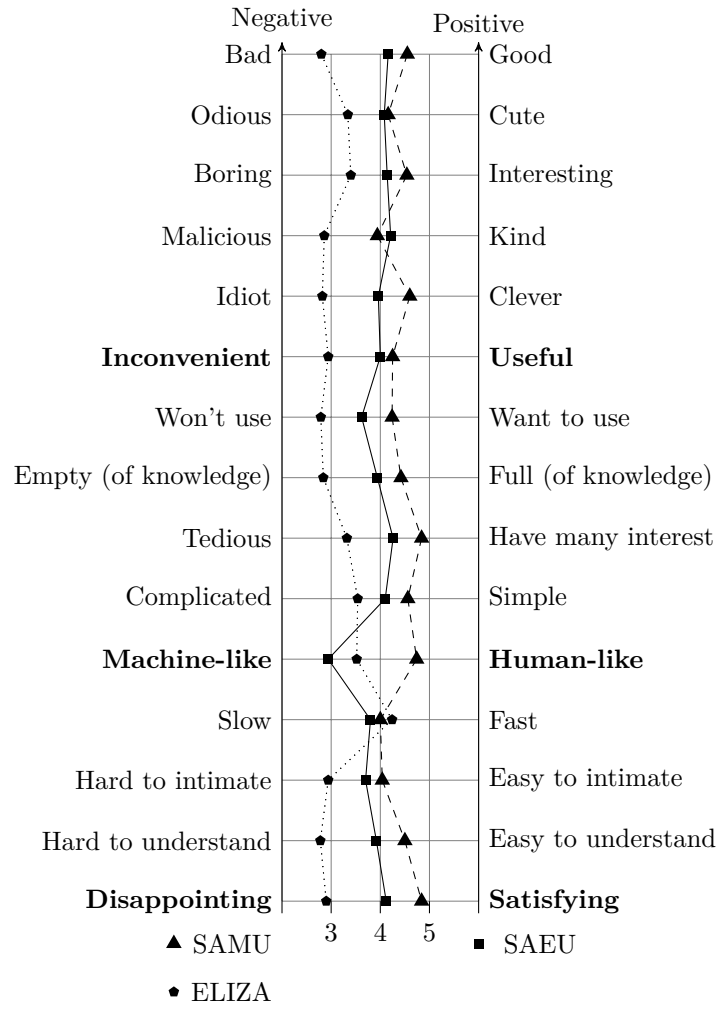


Figure 2.31: Semantic differential experiment's results

Table 2.8 shows the result of the response's evaluation.

Table 2.8: Evaluation of the answers

	SAMU [%]	SAEU [%]	ELIZA [%]	Human [%]
Human-like	82.0	18.0	52.0	88.0
Having knowledge	90.0	94.0	22.0	90.0
Natural	78.0	32.0	36.0	94.0
Unnatural	16.0	64.0	28.0	6.0
No meaning	6.0	4.0	36.0	0.0

Results analysis

First, I confirmed that SAMU's responses are *human-like* (82.0%), *natural* (78.0%), *contain knowledge* (90.0%), and that SAEU's responses are more *machine-like* (82.0%) and a little *unnatural* (64.0%). However, the subjects *understand* them (96.0%) and think they *contain knowledge* (94.0%). The ELIZA's responses are quite *human-like* (52.0%), but often the subjects did *not understand* them (36.0%) or thought they were *unnatural* (28.0%) and in general they contained *no knowledge* (88.0%).

Second, I confirmed that the two new systems give a better impression than an ELIZA system²⁶. Even if ELIZA's answers are quite human-like, the users expect a machine to provide them knowledge and not only to reply with a natural response. In addition, both systems' results are quite similar. Even if the SAEU's responses look more machine-like (the difference is 1.80 points), the other parts of the impressions are just a little worse (the mean difference is 0.34 points). In consequence, I confirmed that human-like answers are maybe not that important for a machine, even if responses are more machine-like the final impression becomes just a little worse. I think most people understand that a system does not speak like a human and do not really care about that. The most important thing for them is to obtain an understandable response which contains the knowledge they expect.

The three last questions' results give us similar results as the semantic differential experiment's results. In general, the subjects *want to use* SAMU (80.0%), I think it is because it is more human-like. However, if they are asked which system they *want to buy* and which system they think *is useful* SAEU obtains results close to SAMU, I think that is because SAEU looks more objective or precise.

2.4.4 Conclusion

I checked that the same framework can be used to create, many different dialogue agents which have different behaviors, very easily with a minimal time cost. Moreover, if the framework is enhanced all the systems using it will be impacted at the same time.

The results from both systems are better than an ELIZA system and give a good impression to the user. As expected, SAEU was more machine-like, but that did not really annoyed the subjects. In general, most of the users did not expect a machine to reply like a human and forgave a machine that is not natural if the response was understandable and contained the expected knowledge.

2.5 Conclusion

I proposed three different languages-specific rule-based systems targeting the Japanese language. I compared different possible behaviours and checked that all of them have advantages and drawbacks. However, the proposed systems are not enough flexible to handle all the possible inputs, even if they have knowledge to reply to the input, they often cannot match a particular rule to generate an output. In addition, some errors of

²⁶SAMU and SAEU means are 4.41 and 3.93 instead of 3.10 for the ELIZA system

the morphological analysis tool influence the system responses and the system was not able to acquire correctly some inputs which are not grammatically correct.

Therefore, I considered the proposed systems have a good learning capability. However, it was insufficient to acquire enough knowledge and to reply to all the user inputs. Moreover, since the systems focus on sentences, it would be complicated and time expensive to make this kind of systems able to handle inputs which are not sentences such as emotions or environmental information.

In addition, the proposed systems cannot be easily adapted to other languages, since they use language-specific external resources, namely *morphological analysis tool* and *Aiduchi*. In addition, I consider that to be able to have a conversation with a user, a system has to learn the user-specific language. Even Japanese speakers do not speak exactly the same language, each speaker has his own expressions and grammatical constructions. Therefore, I consider a system has to be able to learn any language, i.e. any user language, as well as any language in the common definition of the term such as English, Japanese or French.

That is why after developed several *language-dependent systems*, I considered developing a *language-independent system* in order to improve the learning capability of dialogue agents. Moreover, I thought developing a new system from zero would be a good solution to avoid many inefficient methods and language-dependent methods which cannot be used in more complicated dialogues.

Moreover, the proposed rule-based systems do not check the consistency of their outputs. For example, if the user asks “Do you like cookies?” and “You don’t like cookies?” the system can answer “yes” to both questions. In order to keep consistency, a dialogue agent has to keep an internal state and check if the generated output is compatible with it or not. In addition, context also has to be saved to avoid other inconsistent outputs. Nevertheless, I considered rule-based systems not suited to handle such internal state since they can not easily represent all the details of contextual information. That is why the newly proposed language-independent system uses another representation.

Chapter 3

Language-independent agent

3.1 Introduction

It is natural at first hand to focus on only one language to develop a dialogue agent. Since it permits to perform many language optimizations and processing adaptations, for example, in English a system can use the question marks to identify questions very easily or in Japanese a system can use different type of character¹ to classify words. However, as I could check with my previous systems (cf. **Chapter 2**), language-specific systems are excessively related to the tools they use such as dictionaries or morphological analysis tools, and as a results are not able to efficiently handle new expressions which are not known by these tools. Therefore, their learning capability is intrinsically limited. That is why I developed a *language-independent dialogue agent* which is not limited by external tools and can learn any kind of inputs using the same algorithm. Precisely, in this research, I defined *language-independent dialogue agent* as below:

A language-independent dialogue agent is an agent which can acquire and process any language using the same algorithm.

First, nowadays with progress in research and system improvements, dialogue agents will be able to handle more and more complicated conversations, for example the *Multi-modal Multi-domain Spoken Dialogue System* [40] has been proposed. However, to reach their objectives they use many high-level operations. Consequently, to handle these complicated processes most of these systems use very language-specific tools such as dictionaries or grammatically tagged corpora which often require domain expert knowledge. For example, many systems [41, 42] including systems I previously proposed, work only in a specific language, such as Japanese, because they use a language-specific morphological analysis tool such as JUMAN to acquire a list of the words contained in the user's input. That is why such systems cannot be easily adapted to another language without a lot of work; morphological analysis tools are not available for all languages and the format of their results can be very different. On an other hand, systems focusing on particular

¹ *Kanji, hiragana, katakana and romaji*

tasks often use more grammatically-related concepts to, for example, answer question. In instance, the adaptation of the *Watson system* to the French language is estimated to several millions of euros², the same adaptation is planned to Japanese and Spanish for a similar cost. That is why I thought it is necessary to develop a *tabula rasa* system in order take in account all the constraints of a language-independent dialogue agent from the starting of the development, like done for the *ANNABELL model* [43].

Nevertheless, some studies have aimed to develop multilingual sentence models [44, 45] which can suit language-independent systems. However, since languages have particular characteristics that are not common to all the languages, such models have to handle many different concepts. For example, the Japanese language uses particles and the English language uses articles, which are mutually not used and need specific representations. That is why the resulting models are complicated and have to be updated for each new supported language. In addition, many methods using examples corpus can be applicable to any language, but they at least require the list of the words present in the sentence [48, 49]. Some methods are applicable to any language like, for example, the *N-gram model*, *artificial neural networks* [46] and the *hidden Markov model* [47]. However, alone they are not enough to create a flexible dialogue agent. That is why I attempted to combine language-independent methods' concepts in a single *graph representation* in order to represent the entire language characteristics using a same simple structure.

Moreover, language-specific systems can use some language-relative tricks to gain better results, such not citing the subject in Japanese sentences in order to allow the user to guess it. However, this behavior is not possible in other languages such as English. In addition, the use of language-related processing means that most systems only accept plain text input, and makes the task of developing a multimodal system complicated and time-consuming, since special algorithms have to be developed for each kind of input. In contrast, a language-independent system can handle plain text as well as any tag representing, for example, the environment of the system similarly.

Furthermore, using external language tools causes most systems to be unable to efficiently learn new expressions or frequently-used grammatically incorrect sentences like it was the case of my previous systems (cf. **2.3**). For example, if the user input the sentence "I smurf today", the verb "smurf" can be not recognized as a verb and make the system enable to reply correctly. Especially, if a system uses a dependency parser such as *Cabocha* [20], unknown words can have a big influence on the parsing result. However, a language-independent system has no restrictions and can learn any kind of input and adapt to the user's speech, even if the input is grammatically wrong or contains new words. Consequently, the system can acquire the user-specific language in order to understand and reply correctly in future dialogues. Therefore, I consider that a language-independent system has important learning potential.

That is why using my previous experience, I proposed the framework of a dialogue agent that has been developed with the aim of handling any kind of language including languages for which no natural language processing tool is provided, which is the case of most of the languages. As a consequence, the proposed system uses no language-specific

²lemondeinformatique.fr/actualites/lire-ibm-watson-travaillera-en-francais-en-2016-62431.html

tools, such as morphological analysis tools or dictionaries, to keep the algorithm applicable to any language for which enough samples are provided. The system only uses language samples, which can be easily created automatically without requiring human task and domain expert knowledge. The system can acquire any kind of information in order to make the system as flexible and extensible as possible.

I hypothesized that instead of using more external tools to enhance the system, limit these tools will help the system to improve its learning component and its flexibility. Since the external tools limitation such as language or used characters will be avoided.

3.1.1 Objectives

It is hard to develop a language-independent dialogue agent, since most of the previously-proposed methods focus on specific languages and need adaptations to be used. In addition, the results of a language-independent system will maybe not be better than the current best systems. However, it would be useful to achieve many different objectives such as those listed below.

- Handling new terms, such as words used by young people
- Creating a dialogue agent for minor languages for which natural language processing tools are not provided
- Foreign language-learning using casual dialogue as training, especially for vocabulary acquisition
- Accepting different kinds of input in order to develop a fully multimodal system

Moreover, since the system uses no external tools, it can be easily installed on a mobile device and works without a network connection. As a consequence, the system can be configured to never send data to a server to ensure the protection of the user's privacy.

Furthermore, the system could also be setup to handle nonverbal input such as sign language. Since the system can handle any kind of input such as gestures, specials tags or texts, it can be considered naturally multimodal. Another interesting exploitation of the natural multimodality of the system would be to input time tags continuously so that the system output will be influenced by these tags, for example, to make it output “good morning” in the morning and “good afternoon” after midday.

3.2 Technical constraints

Creating a language-independent algorithm mainly has the below constraints.

- Identifying sentence sub-parts without dictionary
- Identifying sentence sub-part' category³
- Acquiring relations between sentences and sentence sub-parts automatically

³nouns, adjective, adverb...

3.3 Real-time language-independent dialogue agent

Concretely, the proposed system uses a *language-independent model*, i.e. a language-independent set of links, applicable to any language in order to create a model of the language from samples of the target language. The generated model, i.e. the generated graph, depends on the target language given as input and is language-specific. Furthermore, the system handles simple dialogues and needs to be improved in future to support, for example, input that requires environmental information in order to be replied correctly such as “What time is it?” or “How is the weather?”.

In addition, the generated graph can efficiently be used to save an internal state and context information appending simple scores on each node. Then this score can be used to check the consistency of the system output.

3.3.1 System architecture

The newly proposed dialogue agent generates a graph using language samples and then uses graph traversal to generate and select the optimum responses to the user’s input. As shown in Figure 3.1, the system does not use language-independent information, but acquires and uses the language information, including language-specific words and grammatical constructions, using the same processing, i.e. language-independent processing, in any language in order to provide a dialogue in a specific language. The essential point of the proposed method is being able to learn any language from simple language samples, i.e. samples without language-expert tags, language-dependent tools or annotations (like POS tags), using the same processing.

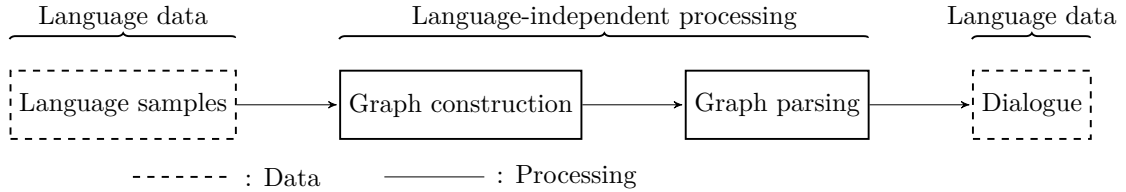


Figure 3.1: Language information flowchart

Concretely, each node of the generated graph represents a sentence such as “hello” or “I am hungry” or a part of a sentence such as “hungry” or a part of word such as “hung” and each link represent a relation between nodes. Figure 3.2 shows the principle of the output generation used by the system. The input of the user is compared to the *input nodes* which are used as entries of the graph, and then if it is similar all the related nodes’ score is increased. Finally, the nodes which have the highest score are selected as outputs. The system’s goal is similar to a search engine, which looks for the best pages corresponding to the user input. Therefore, the proposed approach to choose the output can also be compared to the *PageRank* [50] algorithm, in which each node corresponding to a page and the related page’s score is increased in function of the current pages’ score.

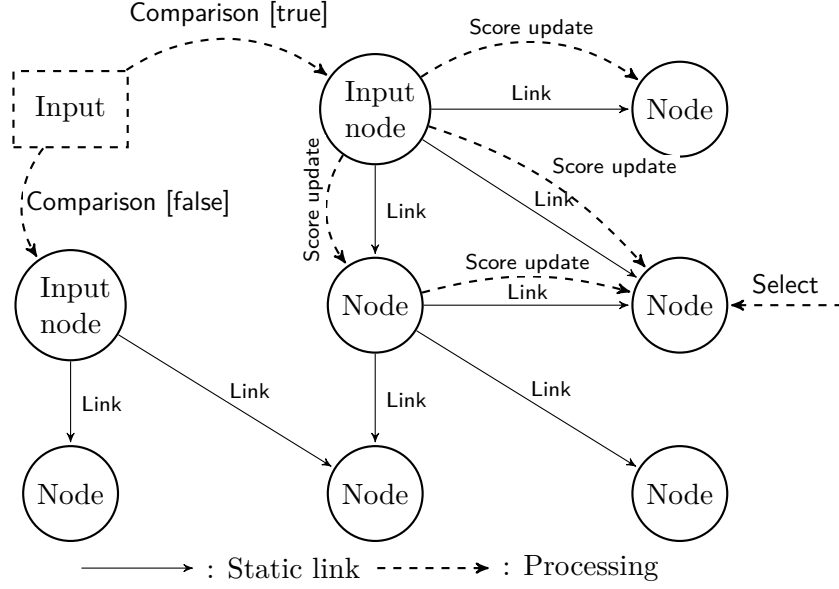


Figure 3.2: Principle of output generation

In another way, the graph can be compared to a *neural network*, used for example for modelling sentences [46]; nodes which can be used together are related by a link.

The main advantage of using graphs is to be able to adapt previously-proposed algorithms to improve the system (cf. **3.3.3**). In addition, this makes it easy to represent how the system works in a human comprehensible way, and makes the system more easily improvable.

The graph is generated using a corpus of example sentences in the target language, and the system output generation is similar to an *example-base dialogue system* [17]. The system looks for sentences that are strongly related to the input and then selects output from them. However, the proposed method does not use any language-expert pattern such as pattern using part of speech or category to select the output; it only uses general methods which can be applied to any language.

Consequently, as shown in Figure 3.3 the system is composed of two main parts: *graph construction*, which replaces the use of external tools, and *graph parsing*, which is used to generate the system's responses.

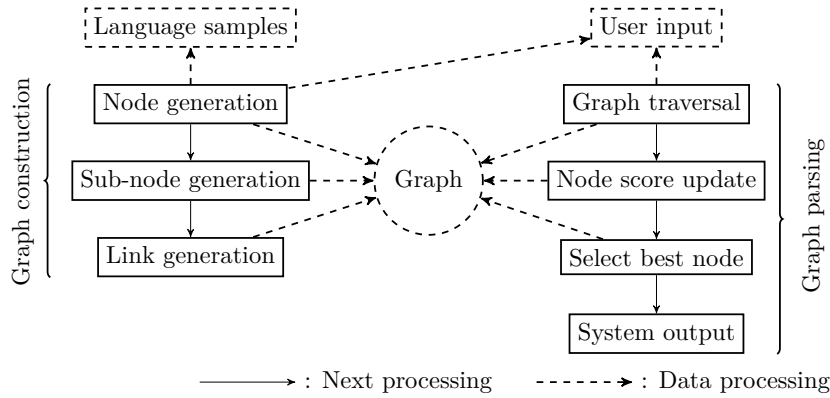


Figure 3.3: Flowchart of the system

In future development, in order to handle more complex dialogues, the first part must be improved to, for example, automatically acquire similar knowledge as knowledge presents in a lexical database such as *WordNet* [21].

3.3.2 Definition of the graph

All the knowledge acquired by the system is represented in a graph using *nodes* and *directed links*, as shown in Figure 3.4. It is important to note that this graph is a very simplified example of a graph used by the system; the graph used by the system contains many more nodes and links.

The graph construction requires only information that is present in dialogues in any language, such as sentence, part of sentence and relation between two elements. That is why the same algorithm can be used to generate graph representing any language. The used links and nodes do not represent language-specific concepts such as parts of speech, but only simple relations present in any language. Concretely, the generated graph is another representation of the samples of the language, it contains the same knowledge, but is more easily usable to generate outputs to an input. It can also be considered a pre-processing of the samples in order to achieve the output generation more efficiently. The current system only acquires basic knowledge. However, the same graph can contain meta-data such as time, emotions or other elements such as gestures or tasks.

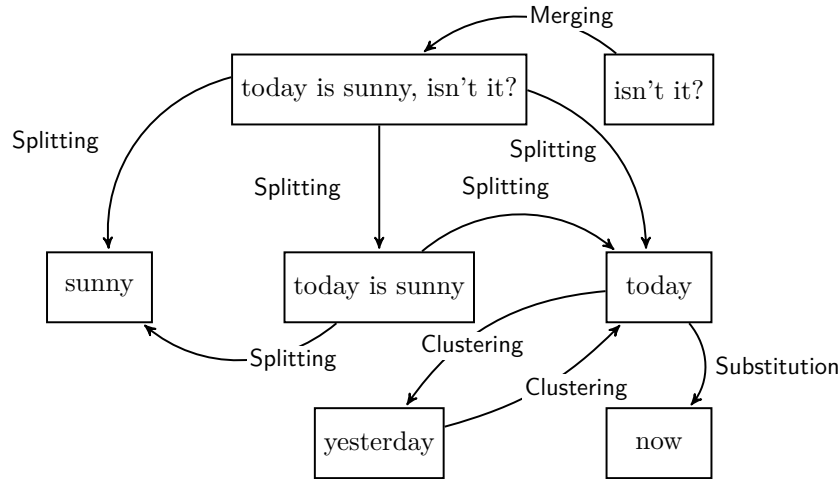


Figure 3.4: Example of a simple graph

3.3.3 Used links

The system uses only *basic links* which are necessary for output generation [51]. These links are a representation of the structure of the sentence and a representation of the semantic relationships between words. Other kinds of link could be added in the future to improve the system output, especially to represent meta-data attached to a node.

Splitting link

A *splitting link* is generated between a node and its sub-node. Here, sub-node refers to a node which value is contained in another node value. For example, the node “I like pears” contains the nodes “I like” and “like pears”, and is consequently related to it. *Splitting links* can be compared to the N-gram model; however, they do not consider the order of the nodes. *Splitting links* are added to a node during the generation of sub-nodes (cf. 3.3.4).

Merging link

A *merging link* is the opposite of a splitting link; *merging link* and *splitting links* are always added together. It is set between a node and its super-node. For example, there is a merging link between “How” and “How are you?” or “How are” and “How are you?”.

Specifically, a *merging link* is set between a sentence’s part sub-node and all the complete sentences which include this sub-node. Using these links, the system can retrieve complete sentences that are eligible for output. They are generated at the same time as the splitting links, however their characteristics are different (cf. 3.3.6).

Substitution link

A *substitution link* is provided between a *node A* and a *node B*, if the *node B* can be used instead of the *node A* in the output. For example, when the user inputs “Hello”, the system can answer “Hello”, “How are you?” or “How do you feel?”. As a result, there are *substitution links* from “Hello” to “How are you?” and to the other possible responses. I consider that a *substitution link* is a similar process to *association* in psychology [52] and it has similarities with the *hidden Markov model*.

However, if the input is “How are you?”, the system may reply “I am fine” and “I am tired” at the same time, which would not be coherent behavior. To avoid this kind of unexpected action, it is possible in a other version of the system to implement emotional concepts (cf. 3.4); the node “I am fine” can be connected to a good emotion, i.e. a node representing this emotion and the node “I am tired” to a bad one, and then the system can be set up to output only nodes related to the emotion corresponding to its current emotional state. These emotional nodes are not related to language, since the same basic emotions are used by all humans [53]. Nevertheless, avoiding more complicated kinds of contradiction such as “I am sleeping” and “I am eating”, would require many information about context to check in what condition both actions can be done and check if they are mutually compatible or not.

In addition, I also distinguish substitutions learned from the corpus called *substitution sentence*, and those between parts of those sentences called *substitution part*. The first is considered to always be correct and the second to be possibly erroneous. This distinction allows the system to define priority during the output generation.

Substitution links are not enough to be able to answer questions efficiently, as the system needs to acquire other links to know which nodes have to be deleted or replaced from the question to generate a correct response. For example, if the question is “What do you like?” the system needs to remove the question mark in the question and replace the interrogative pronoun with a noun to generate a response such as “I like chocolate”. If the system learns which parts of the question to replace and which parts to remove, it will be able to answer many similar questions.

Substitution removing link In many languages, a question contains a component that indicates it to be an interrogative sentence; for example, question marks in English and French or the particle “*か* (*ka*)” in Japanese. The system needs to avoid all the nodes which contain this part in order to not always answer questions with another question. Nevertheless, the same node can be included in a declarative sentence, that is why the system does not completely avoid node containing it but only decrease its score to give priority to other.

The system puts a *substitution removing link* on a sub-node of the question which is not present or replaced in the answer. Concretely, the link loops on the same node. For example, in the case of the tuple “It is hot, isn’t it?” and “It is hot”, the system puts a *substitution removing link* on the sub-node “isn’t it?”.

There are also other expressions dependent on language such as “*est-ce que*” in French, which serves to convert a statement into a yes-no question. These kinds of construction

can be very useful for answering questions naturally. However, they are variable between languages and are hard to handle in a generic way, but with enough knowledge they can be handled by the system like other expressions.

Substitution replacing link In addition, interrogative pronouns have to be replaced and never used in the answer. That is why the system puts a *substitution replacing link* on a sub-node of the question which is replaced in the answer. This link causes the system to avoid output containing the target node. For example, in the case of the tuple “What do you like?” and “I like peaches”, the system puts a link on the sub-node “What”. Using these links, the system can reply to questions in a natural way and avoid responses such as “I like” to the question “What do you like?”.

Figure 3.5 shows an example of a graph containing the different kinds of *substitution link*.

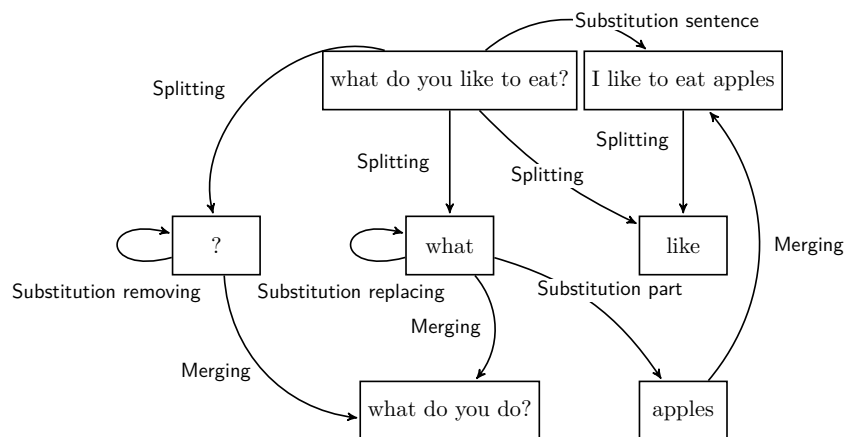


Figure 3.5: Example of a graph containing different substitution links

Association link

Association links represent the degree of similarity of two nodes. They are required to generate the *clustering links*. Two words are related if they appear in the same sentence as another word [54]. These relationships can be discovered by checking the graph structure. For example, if we take the node “I like” as a base, we can go to the sentences “I like apples” and “I like peaches” using a merging link, then using a splitting link we can go to the nodes “apples” and “pears”, which can be considered as related via “I like”, as both of these are “likable things”.

The co-occurrence of words is defined as a path $\{\text{merging link}, \text{splitting link}\}$ in the graph. The system simply has to find this path to look for related nodes.

Clustering link

The *MaxMax soft-clustering method* [55] is used to create clusters of nodes that are related by *association links*. The MaxMax algorithm has been made to suit tasks such as *Word Sense Induction* (WSI) [56]. It is a non-parametrized and graph applicable algorithm which is very easy to implement and usable in different languages [57]. However, other clustering algorithms that work on graphs can be easily adapted to be used in the system.

The system cannot discover the word part of speech to improve the precision of clustering. However, it uses the number of links to classify nodes. For example, the node “I” has many merging links, and the node “prognostication” has only a few links. As a generalization I hypothesize that pronouns or verbs have more links than nouns. The system only considers nodes which have a similar number of links⁴ to create clusters.

In addition, I had to consider the variation of the number of characters. I consider two kinds of language: those with a small number of different characters where each word contains several letters, like French and English, and those with a high number of different characters where many words are composed of only one or two characters, like Japanese and Chinese. The number of substrings is $n(n+1)^2$ (n is the number of characters), and consequently more a language uses long strings, the more sub-nodes may be generated (cf. **3.3.4**) and the more chances the system has to make mistakes while clustering.

Figure 3.6 shows an example of two possible clustering results for words related to “I eat”. The second result is expected to generate correct outputs.

I eat {“apples”, {“apples”’s sub-nodes: “appl”, “ap”, ...}, ...}
I eat {“apples”, “peaches”, “pears”, ...} (*expected*)

Figure 3.6: Example of two possible clustering results

3.3.4 Graph generation

The system uses training samples of the targeted language (cf. **3.3.9**) to generate the first nodes of the graph. Firstly, each sentence present in the samples is converted into an *input node*. For example, the sentences “Hello” and “How are you?” are converted into two distinct nodes. Then, the system also converts the user inputs into new input nodes. For example, if the user input “I like spring”, a new node containing this sentence will be generated.

The generated graph contains the language information, and as a consequence is language-dependent, i.e. a graph generated with English samples contains English sentences, whereas if Japanese samples are used, it will contain Japanese sentences. Nevertheless, since the generation method and the exploitation of the graph are language-independent, the proposed algorithm is applicable to any language.

⁴Two nodes which have a number of links different from more than 30% are not put in the same cluster. For example, a node which has 100 links and a node which has 131 links are not put in the same cluster.

Sub-node generation

In natural language processing, one of the most common tasks is morphological analysis. However, in the context of a multilingual system tools such as *ChaSen* [58] which are only available for some languages cannot be used. A solution would be to use *unsupervised word segmentation* [59]. However, I needed a real-time and fast adaptive algorithm and consequently, I had to develop a specific sentence tokenizer algorithm to be able to split a sentence⁵.

This is why I simply used already existing nodes to attempt to split new ones. For example, the system will use the node “I like” to split the node “I like peaches” into “I like” and “peaches”.

The generated sub-nodes can represent several words, e.g. “like peaches”, as well as a single word like “peaches” or a part of a word like “ach”. This behavior can be useful for handling determiners such as “a” or “my”; for example, “a cat” and “my cat” are different things and can be represented by different nodes.

This method can generate some noise, i.e. nodes that are not useful for output generation, as well as useful ones. For example, in the experiment, the sentence “where is your friend?” has seven splitting links, “you”, “where is”, “where”, “is your friend?”, “your friend?”, “r friend?” and “where is you”. About 2% of generated nodes contain a part of a word in English. However, as has been proven for *stochastic resonance* [60], noise could also help the system to generate correct and useful responses. Specifically, the system may access ineffective nodes which will not be used to generate the output of the system, because they are regularly related to all the other nodes and will not increase the score of a particular node. As a result, their influence on the choice of the output is limited. However, too numerous links make the system slow, that is why I consider forget useless links.

3.3.5 New node inferring

Using the previously-created clustering links, the system is able to create new sentences by substituting a node with another node in the same cluster. For example, if the starting corpus contains the sentence “There is a dog” and the system creates the cluster {“dog”, “cat”, “rabbit”}, it can generate the new sentences “There is a cat” and “There is a rabbit”. This process can be compared to the crossover in genetic algorithms. This is a kind of reflection to infer new knowledge from existing knowledge. In general, the system selects as outputs sentences previously acquired from language samples. However, in order to create more various outputs, I consider that a dialogue agent must be able to also generate new sentence in order to provide more precise output to the user and to adapt its output to the input more precisely.

⁵Spaces are not used to split sentence, since they are not commonly used in all the languages.

3.3.6 Link characteristics and node score calculation

Each kind of link between nodes has its own characteristics. These are used during graph traversal to calculate the *node score* and the *link cost*.

- **Node score** denotes the importance of a node. This value is used to select the best node to output.
- **Link cost** refers to how much power is needed to traverse the link and go to the next node. This value is used to limit the graph traversal.

All links that have a *link cost* exceeding a defined value are ignored by the system. Each kind of link has the following three characteristics.

- **Weight** denotes the importance of the linked node; links that reference directly a possible answer such as a substitution link have a high value.
- **Distance** denotes the difference between two nodes. Splitting links only remove a part of the information; consequently their distance is small, contrary to substitution link which can point to a totally different node.
- **Base cost** is used to calculate the cost of the link.

Changing these characteristics will change the system's behavior. For example, we can make the system generate more sentences⁶, but these will not all be correct. Alternatively, we can make the system take more careful behavior⁷ and only output sentences that are definitely correct.

The system uses Equation (3.1) to calculate the *node score*, and the *weight* and the *distance* are calculated by aggregating the total value of the links used to arrive at this node from the user's input. Nodes which score is higher than zero are called *active nodes*.

I chose this equation in order to increase *score* and then to select nodes that are different from the input, but which are related to it, i.e. the system must avoid outputting the same sentence as the input. In detail, the more nodes are followed, the more the *weight* and the *distance* increase. This is why the exponential function is used to limit the selected nodes to the vicinity of the input node. For example, I want to avoid selecting nodes that have a high weight, but which are quite far from the input and consequently, potentially inadequate as output. If the input is "What do you eat?" the response "I eat bread" must be selected, but the response to the latter "Do you like bread?" must be avoided since it is too far from the input and the user may not understand the relation with the input.

$$S_n = \frac{\sum weight}{e^{\sum distance}} \quad (3.1)$$

- S_n is the score of a specific node

⁶Decrease the *distance value* or the *base cost*.

⁷Increase the *base cost*.

For example, if the *node A* is related by a *splitting*, *substitution part* and *merging links* to the input, then its score will be increased by $(2 + 5 + 3)/e^{0.75+2+1} = 0.24$.

In addition, the system uses Equation (3.2) to calculate the *link cost*. The *base cost* is defined for each kind of link. However, in order to decrease the influence of very frequent sub-nodes the system uses the number of links in a similar way to the *tf-idf method* [61]. Specifically, the more links a node has, the more the cost of each link increases. The objective here is to limit the use of nodes such as “are” or “do” which are present in many sentences. In addition, a logarithm is used to reduce the difference between two nodes that only have a small difference in number of links, and consequently can be considered similar.

$$C_l = c \times (1 + \log(n_{link})) \quad (3.2)$$

- C_l is the cost of the link
- c is the base cost of the target link type
- n_{link} is the number of links of the corresponding type from the same node

Table 3.1 contains the empirically defined values for each type of link. In detail, these values have been chosen after many manual attempts to reach the optimal compromise between the number and the correctness of the outputs. The association and clustering links are used to create new nodes, but are not used during graph traversal, i.e. during the output generation, and as a result do not have specific characteristics.

Table 3.1: Link characteristics

Link	Cost	Distance	Weight
Splitting	1.50	0.75	2.00
Merging	0.99	1.00	3.00
Substitution sentence	2.51	2.00	20.00
Substitution part	2.50	2.00	5.00
Substitution removing	2.50	1.00	-8.00
Substitution replacing	1.00	1.00	-10.00

3.3.7 Output generation

The system updates score of nodes in function of the input, and then selects the highest ones as outputs.

In order to update the score of all the nodes related to the user input, the system accesses the graph by all the *input nodes* at the same time and looks for all the nodes that match, include or are included in the user’s input, if it is the case the node score is increased. Then, the score of all the related nodes is also increased in function of the

followed links. Since only input nodes which are similar to the input are used to enter in the graph, even if the graph is composed of a huge number of nodes, only a part of them will be visiting to generate outputs. Figure 3.7 shows the score update flowchart.

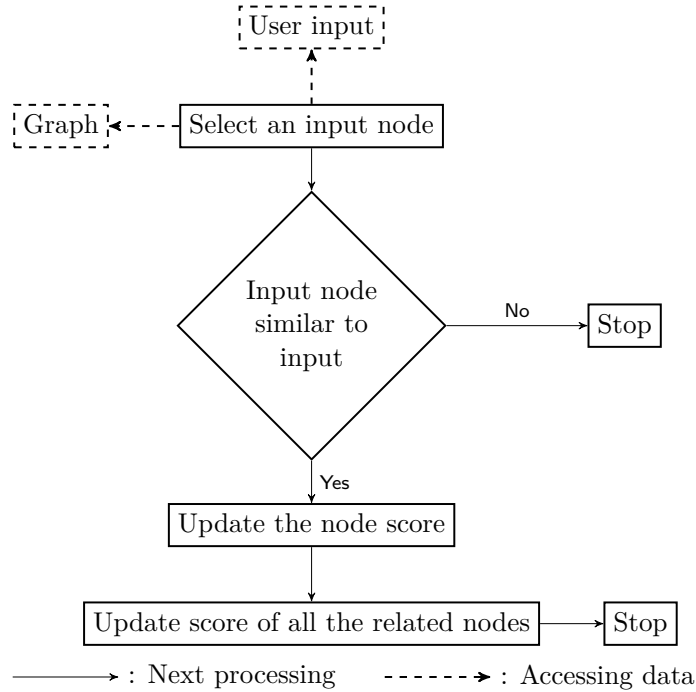


Figure 3.7: Flowchart of the score update

Then, at the same time, the system periodically (every 200 ms in the experiment) checks the graph input nodes to look for nodes which score exceed a *trigger value* (cf. **3.3.8**), then outputs the sentences represented by the nodes to the user. Figure 3.8 shows the node selection flowchart. Since *input nodes* represent only complete sentences, the system never outputs incomplete sentences.

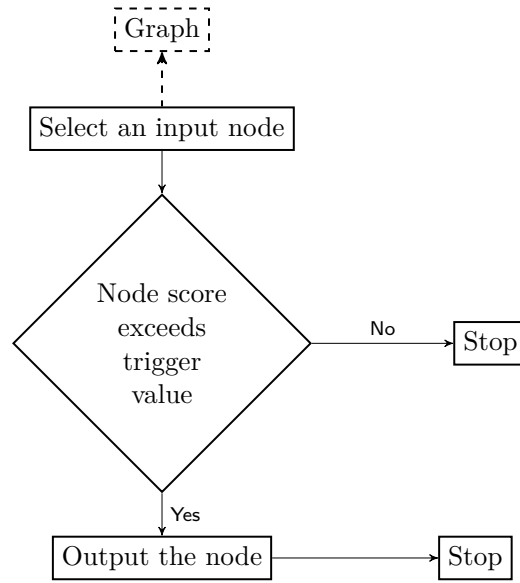


Figure 3.8: Flowchart of the output selection

For example, if the input sentence is “I like making cookies”, the nodes “I like” and “cookies” are included in it, and their score will be increased. Both of these are related to the node “I like eating cookies” by a merging link, and its score will be increased too. If the score of the node exceeds the trigger value, the system will output “I like eating cookies”.

If no node score exceeds the trigger value after the entire graph has been traversed the system will output an *apology sentence* corresponding to a *void tag* (cf. **3.3.8**) such as “I am sorry, I cannot reply”.

Example of output generation

Figure 3.9 shows an example of output generation. In this example, the input is “now, what do you eat?” and the system outputs two sentences “I eat salad” and “I do not eat cookies”. Since the first sentence is closer and its score is higher than the second, the system will output it first. The second response can be considered unnatural, but the system would need more knowledge such as meta-data tags to avoid replying responses that are not totally coherent. We can also note that the score of the input nodes “do you eat?”, and “I do not eat”, “I like apples” increase too, and may be output in function of the parameters of the system.

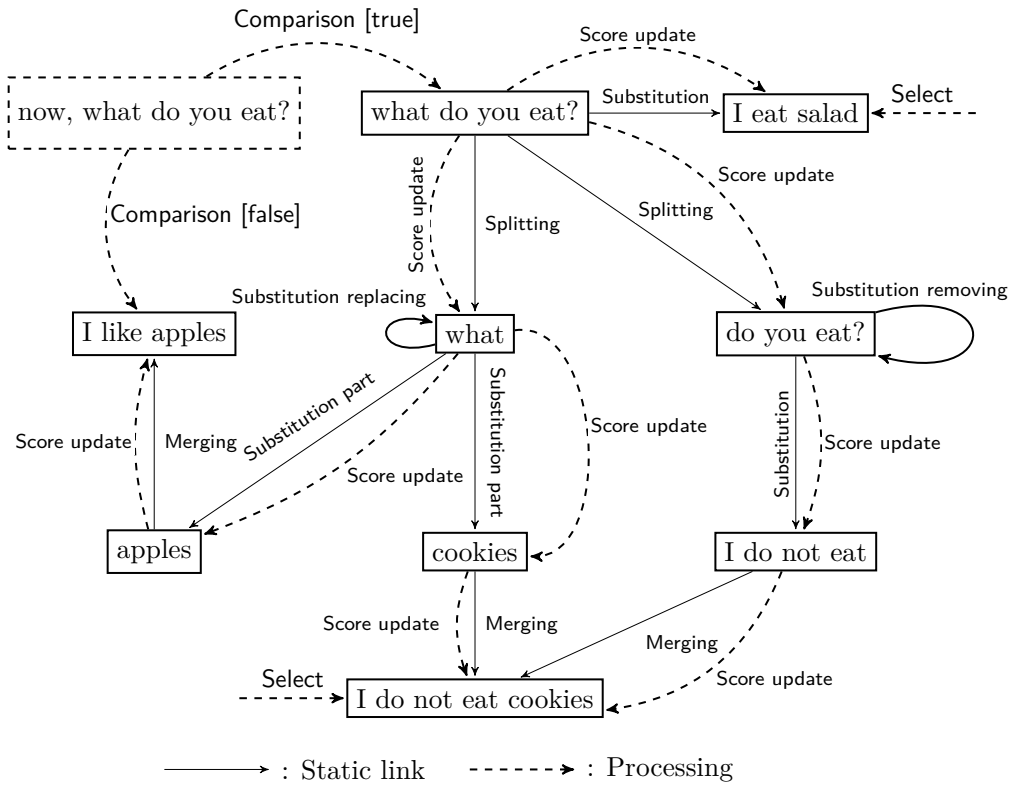


Figure 3.9: Processing of user's input

Example of mistake handling

Using the previously-explained links the system is able to handle some simple *input mistakes* such as misspelling. Concretely, the input string is split into many different substrings, and some of these substrings contain errors, whereas the others are correct and can be used to retrieve a correct string. Figure 3.10 shows an example of a mistake that the system is able to handle, following *splitting link* the system can retrieve correct substrings of the input and then following *merging link* it can retrieve a correct sentence which can be used to generate outputs.

Input I lke chocolate

Splitting {"I l", "ke", " chocolate"}

Merging {"I like"}

Merging {"I like chocolate"}

Figure 3.10: Example of handling of misspelling

In the example, the substring “I l” and “ke” are related to the same string “I like” by a merging link, and since they are contained in it, the score of the node “I like” increases too. Then, because both “I like” and “ chocolate” point to the node “I like chocolate”, this node score is also increased. Finally, the score of the correct sentence “I like chocolate” is increased and the node will be used to generate a response.

It is also important to note that this kind of mistake generally appears when the user uses a keyboard and not a voice input interface.

3.3.8 Implementation of the system

A dialogue is a *real-time process* [32], to enable the system to receive input at any time each operation of the *graph construction* and *parsing* is implemented in a different *thread* which are all executed in parallel. Here, *real-time* means that the system accepts *asynchronous input* and the user does not have to wait for system output to input a new sentence. All the inputs change the structure and the active nodes of the graph, and then influence the output generation.

Figure 3.11 shows the hierarchy of the threads presents in the system. Each thread can access the graph to read or update it.

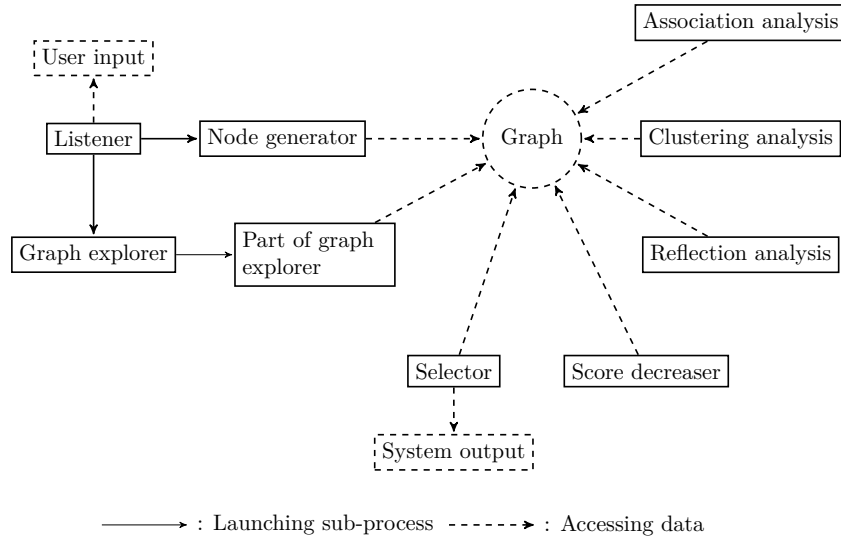


Figure 3.11: System’s threads hierarchy

In order to avoid using all the processing resources each thread is periodically executed. Table 3.2 shows the period of execution of each thread, these value can be adapted to the performance of the machine which executes the system. The *listener* is always waiting user input and as result has a period of 0 ms.

Table 3.2: Threads period

Thread	Period [ms]
Listener	0
Selector	200
Score decreaser	2000
Association analysis	600
Clustering analysis	600
Reflection analysis	600

Association analysis

The *association analysis thread* contains all the processes used to add association links between nodes present in the graph. Specifically, the thread updates the association links of the active nodes using algorithm of Figure 3.12. The input of the algorithm is a node of the graph and it generates *association links* related to it.

```

ALGORITHM generateAssociation(node)
  FOR EACH mergeN OF node.getNodes(Merging)
    //getNodes() returns nodes related by
    //the link in parameter
    FOR EACH splitN OF mergeN.getNodes(Splitting)
      node.addLink(Association, splitN)
      //addLink() adds the defined link to the
      //defined node
      splitN.addLink(Association, node)

```

Figure 3.12: Algorithm used to generate association links

Clustering analysis

The *clustering analysis thread* contains all the processes used to add *clustering links* in the graph. The system updates the clustering links of the active nodes, in order to update the links of nodes that may be used for the output generation.

Reflection analysis

In the *reflection analysis thread* the system generates new possible sentences using the clustering links (cf. **3.3.5**). Concretely, the system attempts to generate new sentences containing an active node. For example, if the node “rice” is active, the system will attempt to generate new sentences using the nodes of the same cluster as “rice”, such as “bread”, to generate new sentences such as “I bought rice” from the original sentence “I bought bread”.

Figure 3.13 shows the algorithm used to infer new sentences. The input of the algorithm is an active node of the graph which may replace a node from the same cluster in an already existing input node. The generated new node is added to the graph in order to be used during the output generation.

```

ALGORITHM generateReflection(node)
  FOR EACH clusterN OF node.getNodes(Cluster)
    FOR EACH mergingN OF clusterN.getNodes(Merging)
      reflection:=mergingN.replace(clusterN, node)
      createNode(reflection)

```

Figure 3.13: Algorithm used to infer new sentence

Listener

The *listener thread* simply waits for the user input. When there is an input it will launch the *node generator* and *graph explorer* thread, to generate the responses corresponding to the input.

Node generator

The *node generator* thread uses the input to enrich the graph used by the system. It uses the input to add new *nodes* including input nodes and *links* into the graph. In addition, the same thread is used to import a corpus into the system.

Figure 3.14 shows the function called when the system adds a new node in the graph. Each time a new sentence is inputted, the system will create a new input node, and then check whether it can be used to split previously acquired sentences, i.e. input nodes. The input of the algorithm is a sentence or a tag from the user or from languages samples. This input is firstly encapsulated (using the *generateNode()* function) into a node, in order to be compared with already existing node, if it already exists no processing is executed.

```

ALGORITHM createNode(input)
  node:=generateNode(input)
  IF alreadyExist(node)
    return
  FOR EACH inputNode OF inputNodes
    //inputNodes is the list of all the input nodes
    checkSplit(node, inputNode)
  inputNodes.add(node)

```

Figure 3.14: Algorithm used when the user inputs a sentence

Figure 3.15 shows the function called to check whether a node can split another and vice versa. The system checks whether a node contains or is contained in the other one,

and if this is the case it splits the containing node by the contained node. In addition, the system will also check whether any sub-node can be used to split the newly-created node.

```

ALGORITHM checkSplit(node, inputNode)
  IF node CONTAINS inputNode
    split(node, inputNode)
  ELSE IF inputNode CONTAINS node
    split(inputNode, node)
  ELSE
    FOR EACH partN OF inputNode.getNodes(Splitting)
      checkSplit(node, partNode)

```

Figure 3.15: Algorithm used to check whether a node can split another

Figure 3.16 shows the function called to split and to set the required splitting and merging links between two nodes.

```

ALGORITHM split(supNode, subNode)
  nodes[] := supNode.split(subNode)
  //subNode is included in nodes
  FOR EACH node OF nodes
    supNode.addLink(Splitting, node)
    node.addLink(Merging, supNode)

```

Figure 3.16: Algorithm used to add a new sub-node to the graph

In addition, this thread will also generate substitution links between nodes it generates. Figure 3.17 shows the algorithm used to create substitution links between a question and its response. The two parameters, *question* and *response*, are nodes representing a part of dialogue of the targeted language samples. Concretely, a *substitution link* is set between an utterance and its response, and a *substitution part link* is set between all the sub-nodes of the utterance and all the nodes of the response which are not present in the utterance. *Substitution replacing links* and *substitution removing links* are also generated when the question contains nodes which are not present in the reply, and as a consequence has to be replaced or removed to select a correct output.

```

ALGORITHM createSubstitution(question, response)
    question.addLink(SubstitutionSentence, response)

    QParts = question.getNodes(Splitting)
    RParts = response.getNodes(Splitting)
    QParts.removeElements(RParts)
    RParts.removeElements(QParts)

    IF RParts.isEmpty()
        FOR EACH QPart OF QParts
            QPart.addLink(SubstitutionRemoving, QPart)
    ELSE
        FOR EACH QPart OF QParts
            QPart.addLink(SubstitutionReplacing, Qpart)
        FOR EACH RPart OF RParts
            QPart.addLink(SubstitutionPart, RPart)

```

Figure 3.17: Algorithm used to create substitution links

Graph explorer

The *graph explorer thread* starts a *part of graph explorer* thread on all the input nodes of the graph. As a result, the graph is accessed by several agents at the same time. This parallel processing improves the speed of the output generation.

Part of graph explorer

The *part of graph explorers* look for nodes that match, include or are included in the input, starting with an input node.

As shown in Figure 3.18, the algorithm takes the user input as parameter, then it is firstly encapsulated into a node to be compared with each input node of the graph, in order to look for all the nodes that match, include or are included in the user's input⁸; all of these nodes' scores will be increased.

⁸For example, the node "I like eating" includes "eating" and is included in "I like eating chocolate". As a result, these two nodes' scores are increased.

```

ALGORITHM visitingGraph(input)
  node:=generateNode(input)
  FOR EACH inputNode OF inputNodes
    IF inputNode = node
      inputNode.increaseScore()
      followLink(inputNode, 0, 0, 0)
    ELSE IF inputNode contain node
      inputNode.increaseScore()
      followLink(inputNode, 0, 0, 0)
    ELSE IF node contain inputNode
      inputNode.increaseScore()
      followLink(inputNode, 0, 0, 0)

```

Figure 3.18: Algorithm used to find nodes related to the input

Then, as shown in Figure 3.19, using the previously acquired links, the score of all the nodes that are related to a matching input node will be increased too in function of their links' characteristics. All the nodes will be accessed until the *link cost* exceeds a defined value, i.e. *MAXCOST* (set to 5 in the experiment). The cost of each link is added to the previous *link cost*, and as a result the cost used in the comparison increases each time the system follows a link.

```

ALGORITHM followLink(node, weight, distance, cost)
  FOR EACH link OF node.getLinks()
    //getLinks returns all the links related to a node
    weight:=weight + link.weight()
    distance:=distance + link.distance()
    cost:=cost + link.cost()
    IF cost < MAXCOST
      linkedNode:= link.getNode()
      linkedNode.updateScore(weight, distance)
      followLink(linkedNode, weight, distance, cost)

```

Figure 3.19: Algorithm used to update score of the node

Score deceiver

Since the system increases nodes score, it is also necessary to decrease score in order to not output node which was in relation with previous inputs.

The *score deceiver thread* causes the score of all the active nodes to fall to zero after a defined time which vary in function of the node score (about 5 seconds in the experiment). Concretely, the nodes' scores are periodically updated using Equation (3.3). The score of each node is divided by a coefficient, k , (set to 2 in the experiment) in order to decrease its score in function of its current value. In addition, one is removed in order to be sure that the score will tend to be a negative value and, as a result that all the nodes are set inactive if there is no input during some time.

$$S_t = \frac{S_{t-1}}{k} - 1 \quad (3.3)$$

- S_t is the node score at time t
- k is used to determine the decrease order

Precisely, the score value is continuously brought closer to zero using a unique parameter, and when it becomes zero, the node is set inactive.

Selector

The *selector thread* looks for nodes for which their score exceeds the current *trigger value* (cf. **3.3.8**). In addition, it also updates the current trigger value in function of the time and the system's previous output.

As shown in Figure 3.20, the node that has the best score, which exceeds the trigger value, and which is a complete sentence, is selected and output to the user. The algorithm does not require any parameter, since it only checks the graph state to look for possible outputs. Outputted node score is set to zero to not output several times the same sentence.

```

ALGORITHM checkOutput()
  currentN:=null
  FOR EACH node OF activeNodes
    IF NOT node.isInputNode()
      continue
    //isInputNode() returns true if
    //it is an input node
    IF node.score() < outputTriggerValue
      continue
    IF currentN = null
      currentN:=node
    ELSE IF node.score() > currentN.score()
      currentN:=node
  updateTriggerValue()

  IF currentN = null && NoResponseToInput
    input(voidTag)
    //A voidTag is input in the system
  ELSE IF currentN != null
    resetTriggerValue()
    currentN.score:=0
    output(currentN)
    //The currentN node is displayed
    //to the user

```

Figure 3.20: Algorithm used to generate output

VoidTag If the system finished exploring the graph and no node can be outputted to reply to the user input (*NoResponseToInput*), then it will input a *voidTag* in the system exactly like user sentences are input.

Some dialogue samples provided to the system as basic knowledge contain this tag and the corresponding response, and consequently the system will be able to provide an output to it such as “Sorry, I cannot answer”. Precisely, the *void tag* is substituted, using a *substitution link*, with an apology sentence to indicate to the user not to wait for any other output.

Output trigger value In order to create a real-time system, the system has to reply in a minimum amount of time like a human would, but with maximum relevance, i.e. with the best possible response. In order to produce this kind of behavior, the system uses a *dynamic trigger* representing the minimal score required by a node to be an output candidate. The initial value is set in function of the system state and then the value of the trigger decreases in function of the time spent, using Equation (3.4). Consequently, the system searches first for nodes that have a high score, and the more time passes, the more it searches for lower-scoring nodes, since the trigger value becomes lower.

$$V_t = V_{i-1} - t \times k \quad (3.4)$$

- V_i is the initial value
- t is the time since the initial value
- k is used to determine the decrease order

This equation makes the trigger value decrease using a single parameter, k , that is empirically set (set to 2 in the experiment). The parameter can be changed to make the system output more or less sentences for each input.

Figure 3.21 shows an example of the trigger changes.

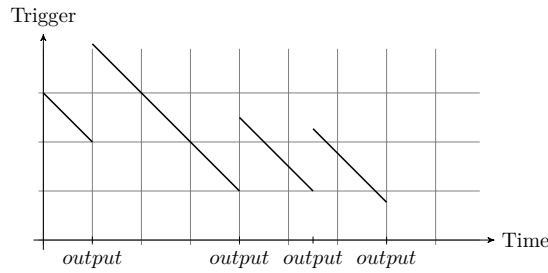


Figure 3.21: Example of trigger changes

After the system selects an output, the trigger value is reinitialized. This value is calculated using Equation (3.5). Concretely, the system uses the mean of the last five

output values ($n = 5$) multiplied by 2 ($m = 2$) to calculate the new trigger value. This method allows the system to adapt to the nodes' scores automatically.

$$V_i = (\sum_{k=i-n}^{i-1} V_k) / n \times m \quad (3.5)$$

- V_i is the value of the trigger at time i
- i is the output number
- n is the number of previous output used
- m is used to scale the trigger value

3.3.9 Training samples

To generate the graph, the system uses two kinds of basic resources, *dialogue samples* and *knowledge samples*. They contain no grammar information and need no complex creation processes. Compared to a common system based on AIML [11] database, the corpus of the presented system contains no tags and all the rules are automatically acquired from the samples. For example, wild-cards, which are often present in corpus-based chat-bots, are not present in the samples. They have to be statistically induced⁹ by the system.

Samples are different for each language and can contain particular characteristics not common to all languages. For example, although metaphors are widely understandable between different cultures [62] (such as “time is money”, “love is madness”, etc.) there are expressions rooted in culture. For instance, *raccoon dog* in Japanese can indicate a cunning person, while in Western cultures this animal has no such connotations. However, since the graph generated from samples and from user's inputs is language-specific; the language model adapts to the user. Therefore, the system will not output utterance “he is like a raccoon dog” in English since it is unlikely to learn it from the user, but it could be learned from a conversation in Japanese. In addition, the lack of distinct difference between single and plural in Japanese is not a problem since the samples and the user's inputs will be adapted to it. For example, number can be defined by specific expressions or by context using relations to other nodes. Even if these expressions are specific to one language they are represented within the same language-independent set of links.

Moreover, it is also possible to create samples which contain only tags indicating sentiment or environment context and which could be used in different languages. For example, the positive sentiment of “good” can be related to “sunny weather”.

In detail, the sentences of the samples are used as arguments of previously-presented functions such as *createSubstitution()* or *createNode()*.

⁹A word that has many substitution links can be considered to be a kind of wild-card.

Dialogue samples

The *dialogue samples* contain some very simple dialogues such as those shown in Figure 3.22, used to acquire *substitution* and *splitting links* in the target language. They can be, for example, extracted from a dialogue between two humans or from screenplays.

U1: What do you drink?
U2: I drink milk

Figure 3.22: Example of a dialogue sample

Language differences Each language has its own way to express some casual questions. For example, during the experiments subjects used four forms of asking someone's age in Japanese, two in French and one in English. Figure 3.23 contains some questions asked to the system by the user in different languages.

- 你多大了 (*Ni duodale*) (*zh*) How old are you? (*formal*)
- 你几岁了 (*Ni ji suile*) (*zh*) How old are you? (*informal*)
- How old are you? (*en*)
- *Tu as quel age ?* (*fr*) How old are you? (*informal*)
- *Quel age as-tu ?* (*fr*) How old are you? (*formal*)
- 何歳ですか (*Nan-saidesu ka*) (*ja*) How old are you? (*formal*)
- いくつですか (*Ikutsudesu ka*) (*ja*) How many are you? (*informal*)
- 年齢を教えてください (*Nenrei o oshietekudasai*) (*ja*) Tell me your age (*formal*)
- 年は? (*Toshi wa?*) (*ja*) What's your age? (*informal*)
- 몇살이에요 (*Myeoch sal-ieyo*) (*ko*) How old are you? (*informal*)
- 너몇살이야 (*Neo myeoch sal-iya*) (*ko*) How old are you? (*formal*)

zh: Chinese^a, en: English, fr: French, ja: Japanese, ko: Korean

^aMandarin Chinese

Figure 3.23: Different ways to ask one's age in different languages

A dialogue system has to be flexible enough to reply to all of these different forms using limited samples. In order to handle this problem, the proposed system can, for

example, use the sub-node “몇살” (“*myeoch sal*”) of the Korean sentence “몇살이에요” (“*Myeoch sal-ieyo*”) to output the response to this sentence when the user inputs “너몇살이냐” (“*Neo myeoch sal-nya*”) too. Consequently, the system can correctly output many variations of questions that are not present in the samples, using sub-nodes of questions that are in the samples.

Knowledge samples

The *knowledge samples* are a list of simple sentences such as “I like ice-cream” or “The president of the USA is Obama”.

The knowledge samples are used to increase the possible outputs of the system. These samples can be collected easily, since they consist of a list of simple sentences that are not contextually related. They can be collected from any text such as a Wikipedia article, a book or from users’ inputs.

3.3.10 Experiments

In order to check the validity of the proposed method, I carried out output correctness experiment. I used the same protocol as the evaluation of generality of SeGa-ILSD [14]. I chose this baseline since it focuses on casual dialogues and has been adapted and evaluated in several languages. However, for this system I did not use an ELIZA system to generate part of the responses as did the baseline, since I considered it is too language-dependent. I also did not use morphological analysis tools as the baseline system does, the generated sub-nodes are used instead (cf. 3.3.4).

In order to be able to carry out a correct comparison with the baseline, voice recognition is used as the input method. However, it is important to note that voice input methods use a huge amount of language-related resources and are only available for a small number of languages. As the presented system is text-oriented, during the experiments voice input is simply used instead of a keyboard.

The *Google speech recognition* is implemented in an Android¹⁰ application to acquire the user’s input. This recognition tool uses a speaker-independent *GMM-HMM model*¹¹ [63].

I asked subjects to evaluate the system’s responses as below, where an example response is provided for each choice.

- **Correct reply** Meaning is correct, and expression is natural.
 - **Example** What do you like?: I like playing basketball.
- **Semi-correct reply** Meaning is correct, but expression is not natural.
 - **Example** What do you like?: I like play basketball.
- **Erroneous reply** Meaning is not correct.

¹⁰<http://developer.android.com>

¹¹Gaussian Mixture Models-Hidden Markov Models

- **Example** What do you like?: Apple are ta.

The aim of the evaluation is to check whether the system’s responses are grammatically correct and correspond to the user input. Nevertheless, I asked the subjects to evaluate the system’s response as erroneous if the system does not reply to the input question. For example, if the input is “What will you do tomorrow?” and the response is “I don’t know”, it is considered to be erroneous even if the output is grammatically correct and a human could reply in such a way. Since this response is provided when no node matches the input.

The system *precision* is defined as the rate of responses that are not erroneous.

Baseline

The Se-GA-ILSD [64] is used as baseline. The baseline is a generalization of the GA-ILSD algorithm to several languages. The part of the system which uses genetic algorithm is called *Baseline 1.1*. The ELIZA-like system is called *Baseline 1.2*.

The baseline is not language-independent, but it is a dialogue agent which has been adapted to several languages. In order to do a crossover of two rules, the baseline needs to identify each word in the sentences, for which it uses a morphological analysis tool¹².

Moreover, the baseline uses *Microsoft Japanese recognizer* (Version 6.1), *Microsoft English Recognizer* (Version 5.1) and *Microsoft Simplified Chinese Recognizer* (Version 5.1) as speech recognition tools¹³.

Experiment preparation

Three native speakers each of Chinese, English, French, Japanese and Korean were asked to imagine a simple casual dialogue of about 40 sentences in order to create the dialogue samples.

The Japanese knowledge samples were directly extracted from the samples created for the experiment of *knowledge sharing* on language-specific dialogue agent (cf. **2.4.3**). For this experiment subjects were asked to teach some common knowledge to a spoken dialogue agent. The same samples were also manually translated into the four other languages by a native speaker.

For Chinese (Mandarin) I used simplified Chinese characters. I did not make any distinction between different kinds of English or French.

Splitting parameter To avoid the generation of many nodes¹⁴ in languages using Latin characters, I set a minimal character length of four to split a string in English and French. Moreover, I set a minimal length of two¹⁵ for Korean.

¹²JUMAN Version 5. for Japanese, Apple Pie Parser Version 5.9 [65] for English and ICTCLAS for Chinese [66]

¹³The version 6.1 stems from Microsoft Office 2003 and the version 5.1 is extracted from the package Microsoft Speech SDK 5.1: <http://www.microsoft.com/en-us/download/details.aspx?id=10121>

¹⁴A node for many short strings that can make the system slower and use memory.

¹⁵Hangeul characters are composed of 24 consonant and vowel letters in groups of two, three or four.

The system also did not use capital letters, in order to increase the node matching rate. For example, in the sentences “Cats are cute” and “I like cats”, “Cats” and “cats” are the same word, but they will be considered different by the system because of the capital letter. However, meaningful capital characters are kept, such as in the case of proper nouns.

It is important to note that the knowledge required to know whether a word needs a capital letter or not depends on the language. With a bigger corpus, I think this task can be avoided without significant impact to the system, since the number of nodes will be sufficient to split all the words with and without a capital letter.

Characters depend on the language, but they do not make the system language-dependent. The user can input any character into the system; the output generation process will not be affected. For example, a word such as “t%&3=f” can be learnt by the system in the same way as all other words.

Experiment settings

I asked each subject to evaluate approximately 50 output sentences by the system. I also gave instructions to each subject for them to input only simple, complete and grammatically correct sentences into the system. Concretely, most of the subjects input some greetings and simple questions about the system, such as its name or age, and some more general questions such as “What are you doing?”.

The steps of the experiment are listed below.

1. Choose the language
2. Enter a name
3. Enter a sentence using voice recognition
4. Evaluation of voice recognition (erroneous recognition results are also input in the system)
5. Evaluation of responses to the input sentence
6. Return to step 3

Table 3.3 shows information about the experiment subjects.

Table 3.3: Subject’s information

	Zh	En	Fr	Ja	Ko	Total
Subject	7	4	4	13	5	33
Male	2	2	4	5	1	14
Female	5	2	0	8	4	19
Student	7	2	2	9	5	25
Worker	0	2	2	4	0	8
Age [years]	21.9	21.5	30.0	23.1	22.0	23.1

Table 3.4 summarizes information about the system knowledge. The number of nodes refers to the number of nodes created before the user starts the dialogue. To count the number of words, I simply split words using spaces for English and French, for Japanese I used the *JUMAN morphological analysis tool*, for Chinese I used *Google Translator’s*¹⁶ *word segmentation*, and for Korean I used the *HanNanum tools* [67]. In addition, for Korean, I also checked the result using *Google Translator’s word segmentation*, obtaining 1,296 words. The number of dialogues corresponds to the number tuples question/response taught to the system.

Table 3.4: Corpus information

Number	Zh	En	Fr	Ja	Ko
of nodes	1410	715	839	1623	664
of words	1246	1029	1343	1268	708
of characters	1818	4853	7465	2514	3383
of sentences (knowledge)	100	100	100	100	100
of dialogues (dialogues)	88	69	102	85	107

Experiment results

Speech recognition precision The speech recognition does not recognize punctuation. Consequently, subjects were not able to input commas or question marks.

The precision is calculated using Equation (3.6)¹⁷.

$$P = \frac{n_c}{n_c + n_i} \quad (3.6)$$

- P is the precision

¹⁶<https://translate.google.com/>

¹⁷The common precision formula [68] is used.

- n_c is the number of correct recognitions
- n_i is the number of incorrect recognitions

This formula is the same as that used in the baseline. It is preferred to the *word error rate* (WER) value in terms of capability to carry out a correct comparison with the baseline.

Table 3.5 shows the precision achieved for each language during the experiment.

Table 3.5: Speech recognition precision

	Zh [%]	En [%]	Fr [%]	Ja [%]	Ko [%]	Mean [%]
Precision	93.4	86.7	88.0	86.7	91.7	89.3

Evaluation of responses Figure 3.24 shows an example of dialogue with the system. Evaluations by the subject for each output are presented in parentheses.

U: How old are you?
 S: I am one year old (*Correct reply*)
 U: What is your favorite food?
 S: I like spring (*Erroneous reply*)
 S: I want to eat fried foods (*Semi-correct reply*)
 U: I like fried foods too
 S: I want to eat fried foods (*Correct reply*)
 S: I can eat spicy foods (*Semi-correct reply*)

Figure 3.24: Example of dialogue with the proposed system

We can notice that the response “I like spring” was evaluated erroneous by the subject. The system had no sample similar to “What is your favorite food?”, and as result, outputs the node “I like spring” which was related to the sub-node “favorite” of the input. The semi-correct response “I want to eat fried foods” was selected from the sub-node “food”.

Some subjects also input more complex sentences, for example, a subject inputted the sentence “okay how about we meet on Friday night at 7 p.m.” and the system replied using previously acquired sentence “Yes I am free on Friday nights” which was evaluated *semi-correct*.

Table 3.6 contains the experiment results compared to the baseline [64]. For the baseline, the values in parentheses correspond to the evaluation results for the ELIZA system’s outputs. All the baseline results were obtained in a single evaluation run.

Table 3.6: Experiment results

Language	Correct [%]	Semi-correct [%]	Erroneous [%]	Precision [%]
Chinese	25.9	17.0	57.1	42.9
English	39.1	14.2	46.7	53.3
French	33.9	9.9	56.1	43.9
Japanese	31.7	16.2	52.1	47.9
Korean	34.4	23.1	42.6	57.5
Mean	33.0	16.1	50.9	49.1
Baseline 1.1 (Baseline 1.2)				
Chinese	25.6 (16.0)	13.6 (30.4)	4.4 (10.0)	39.2 (46.4)
English	4.0 (8.4)	16.0 (53.2)	15.2 (3.2)	20.0 (61.6)
German**	17.6 (19.2)	19.6 (20.0)	13.6 (10.0)	37.2 (39.2)
Japanese	14.6 (38.2)	2.5 (13.5)	8.9 (22.3)	17.1 (51.7)
Mean	15.4 (20.4)	12.9 (29.3)	10.5 (11.4)	28.3 (49.7)

Baseline 1.1: Spoken Dialogue Method Using Inductive Learning with Genetic Algorithm (GA-ILSD)

Baseline 1.2: ELIZA-like system

** Baseline 1's experiment has also been carried out in German.

Analysis of results

It is clear that the results are similar in the five evaluated languages, and that they are superior to the baseline results if ELIZA's responses are not counted. In addition, as the results of the baseline are quite different for each language, I consider that the baseline algorithm is more suited to some languages than others.

I also asked each subject for their impressions after completing the experiment. Some users were impressed by interesting responses or somewhat strange responses, whereas others were impressed by the fact that the system asks some questions by itself.

However, the system can still only handle simple input such as greetings and simple questions about the system's personal information or tastes. Nevertheless, the system learns from user input in order to be able to reply with a greater variety of output the longer the user chats with it.

Contrary to the baseline, the proposed system does not only look for rules that match the input, but deconstructs the input into different parts to access all the related terms of the sentence, and finally chooses an output. For example, if the input contains the word "apple", since "apple" is related to "fruit", the system can respond using the word "fruit" instead of "apple". This behavior is very useful for handling problems related to synonyms and paraphrases. That is why I consider the graph structure suited to handle relation between words and to select the best response to an input.

Used resources The baseline uses a morphological analysis tool and an ELIZA system, both of which are language-specific. In particular, morphological analysis tools use many language-dependent resources such as dictionaries and grammatical rules. However, the other parts of the output generation do not depend on language. Consequently, the system can be adapted to other languages with a minimal amount of work for any language for which such tools are provided. However, if one of these tools is not available, the adaptation task becomes much more complicated.

In comparison, the proposed approach only needs target language samples to be trained and then be able to handle a dialogue in the targeted language. In addition, certain processing steps such as the splitting parameter and the removal of capital letters have been added to improve the system’s speed, but these are not necessary for the functioning of the system.

To achieve a fully end-to-end language-agnostic dialogue system, it is possible to start the system without any knowledge and to allow it to acquire knowledge from the user input. However, in this case the supervising process can be annoying for the user. A better method would be to make the system assist with a dialogue between two humans and acquire knowledge in a similar way to a child hearing people around him and finally becoming able to speak. The dialogue samples used can be considered to be dialogue heard by the system during its “childhood”.

Differences between languages The system’s behavior is quite similar in all the tested languages. However, depending on the language, the system tends to answer with a question or to match the wrong knowledge. Especially in French, questions that contain “*est-ce que*” are often replied to with another similar question. *Substitution replacing links* are made in order to avoid this kind of behavior. However, the construction of this kind of question is not the most frequent in the corpus; as a result, it is hard for the system to acquire the correct *substitution replacing links* and to decrease the score of the question’s nodes when the input is also a question.

3.3.11 Conclusion

The newly proposed system uses a unique *language-independent algorithm* to provide a *real-time dialogue agent* to the user. I carried out experiments in Chinese, English, French, Japanese and Korean and obtained similar results in all these languages. Moreover, the precision obtained is superior to the baseline if ELIZA’s responses are not counted. However, the results have to be improved to achieve its different objectives.

The baseline system handles several languages; however it needs to be adapted to each one. In contrast, the proposed system needs no special work to be adapted to another language. For example, we can input into the system both Chinese and Japanese training samples at the same time, and the system will be able to output Chinese as well as Japanese sentences.

In future, I consider that most natural language processing tools such as morphological analysis tools and part-of-speech tagging tools will be able to handle any kind of language

using statistical methods that are not language-dependent. Therefore, these tools could be used to improve this system’s performance. However, while external tools have many advantages, they often limit the flexibility of a system and make it heavier. Moreover, it is hard to conserve knowledge consistency between all the different tools. For example, the system has no reason to know that “high school” is composed of two words if “high” was always associated with “school” within all the knowledge it has acquired, but external tools could separate them.

Moreover, to be able to handle more numerous useful nodes without generating too many useless links and nodes, the system can *forget* the latter. This can be done in a similar way to humans who forget wrong or non-meaningful information. As a result, the system will not require more processing than system using external resources which contain only meaningful data.

In addition, the system acquires knowledge from samples and from user inputs, and as a result can learn language-specific expressions. However, since the system cannot use language-specific operations to analyze the inputs, it needs more knowledge to acquire them in a statistical way. For example, by using part-of-speech tagging it is possible to give more importance to nouns (such as “city”) than to prepositions (such as “to”), but since a language-independent system cannot use part-of-speech tagging, it has to check the frequency of each word in a set of sentences to determine their importance. The system needs to look at the context of each node to use them correctly, while a language-specific system can use external tools to check the use of each node. Consequently, the proposed method requires many environmental information to select the appropriate output.

Finally, since the system does not use any grammar-specific processing, simple sentences, complex sentences and compound sentences are all processed similarly. That is why I consider the structure of the proposed system can be used to handle complex inputs that other dialogue agents cannot handle correctly.

3.4 Implementation of emotional tags

Language-independent dialogue agents are well-disposed to handle non-textual input since they can acquire any input even if it does not represent a conventional sentence. In addition, I consider the meaning of many parts of sentence cannot be only represented with links to other parts of sentence, but requires *meta-data*. These data can represent the environment or the context of the sentence; therefore it can also represent emotions.

That is why on the first hand, in order to increase the precision of the system, I chose to add a simple *emotional processing*. Emotional nodes could be automatically generated and added to the graph using an external resources [69]. However, in order to keep the system language-independent and to carry out comparison with the previous system, I preferred acquiring emotional nodes in a similar way as other inputs in order to keep the system as generic as possible.

Moreover, some methods has been proposed to predicting emotion of the user [70] and adapted to dialogue agent [71, 72]. However, they used specific algorithm to acquire and process emotions, and as a consequence, emotions are not processing like other inputs.

Precisely, I kept the same base as the previously presented language-independent dialogue agent and make some changes and improvements. In the next sections, I explain the differences with the previous system. Concretely, nodes representing emotions are added in the graph and related to each node representing a sentence or a part of a sentence. At the beginning, each node is related to a *NoEmotion node*. Then, using the user input to system infers the emotion related to each node. For example, if the system output is “Today, the weather is good” and the user response is “That is nice”, the emotion related to the sentence or to the part of the sentence “That is fine” (*good emotion*) will be set to the sentence “Today, the weather is good”. The system does this process for each output, and finally if several users evaluate several times the same node, I consider the system can relate a relatively precise emotion to each node.

3.4.1 Links

Firstly, I made some minor changes on the links present in the graph used by the system.

Splitting link (splitting control)

Contrary to the previous system which set a minimal string length to each language in order to limit the number of generated sub-nodes representing short sub-strings (cf. **3.3.10**). The upgraded system uses a *black list* of strings that must not be generated as a sub-nodes, and avoid the generation of all the nodes which are more frequent, i.e. more often included in the *input nodes*. Nevertheless, the blacklisted nodes can be learned by the system as *input node*.

For example, I wanted to avoid the generation of the node “sch” which is just a part of word and has no particular meaning, as a consequence I put it on the blacklist. Then, the system counts the number of input nodes which contain it and all the other blacklisted strings, i.e. counts their frequency, and then avoid the generation of all the sub-strings which are more frequent than the meaning frequency of all the blacklisted strings.

I considered that short sub-strings of words are more frequent than meaningful words. However, as a result, some single very frequent words such as “I” or “an” are not represented by a single node, but included in a longer one such as “I eat” or “an orange”. Nevertheless, I consider this is not a problem, since they not have enough meaning to be used alone.

This *splitting control* is necessary especially in language using limited number of letters and long strings such as English and French, in order to avoid the generation of sub-nodes for all the possible sub-strings, like a node for each different letter. Since this process depends on language samples given to the system, it is considered language-independent.

In addition, using this control, the splitting results may be a little similar as those obtained from a *Bayesian unsupervised word segmentation algorithm* [59].

Substitution link

In order, to improve and to simplify the substitution process, I made a new hierarchy of *substitution links*.

Substitution sentence link *Substitution sentence* is simple substitution link between a sentence and another sentence which can be used as a response to it. For example, the system generates a *substitution sentence link* between the nodes “How are you?” and “I am fine”. Concretely, these links are acquired from the *training samples* and from the *user inputs*, and are considered to be surely correct. As a result, their influence on the output generation is high.

Substitution follow link

Substitution follow links are set between a sentence and a sentence which can be used after it by the same speaker. For example, after someone says “Today is Sunday”, he could say “I want to go out”.

Like *substitution sentence links*, they are acquired from the *training sample* and the *user inputs*. However, their influence is less important to let the prominence to the latter.

Substitution removing link *Substitution removing links* loop on a node which marks a question and has to be avoided in the response. For example, the node “what” is generally¹⁸ presents in a question and has to be replaced by another node in the response.

Substitution removing links are important to make the system have a natural behaviour and avoid replying questions using other questions. For example, if the input is “What do you like to do?”, the system will tend to visit and to increase score of node such as “What do you” and “What do you do?”, and then use them as outputs. However, since the *substitution removing links* tend to make this kind of node score decrease, the system will use nodes such as “I like to play tennis” to answer the input, which is considered a better response.

Substitution part link In order to make a flexible dialogue agent, the system also attempts to generate substitution between sub-nodes of sentences and their responses. Such as for example, between “where” and “at station” from the dialogue “where are you?”, “I am at station”. Since the system generates many of those links, even if many of them are not meaningful, they can be a useful statistical indication. As a consequence, the importance of each single *substitution part link* is small comparing to other substitution links.

Figure 3.25 shows an example of graph containing the different kinds of substitution links.

¹⁸There are many exceptions such as in the sentence: “This is what is like”.

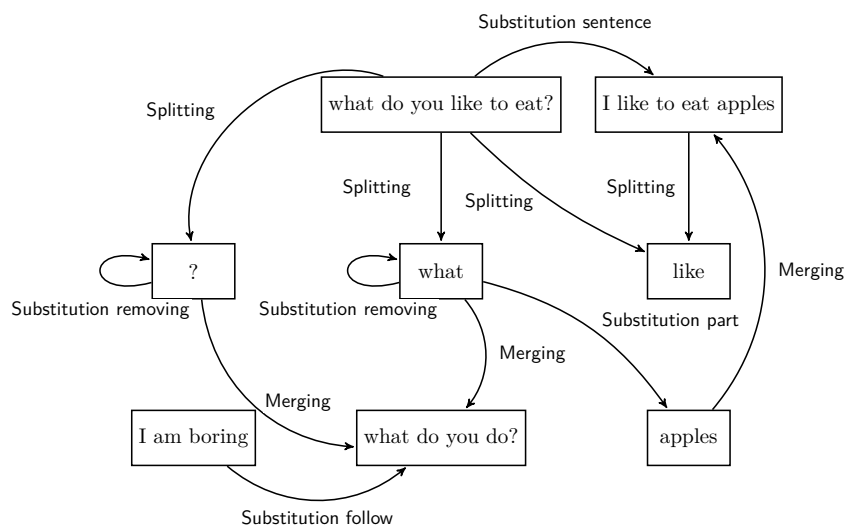


Figure 3.25: Example of a graph containing different substitution links

Clustering link

The system uses the same clustering algorithm as the previous system (cf. **3.3.3**). However, in order to increase its precision some constraints have been added to generate a link between two nodes.

Two nodes can be in the same cluster only if they are bordered by the same nodes. For example, “apple” and “orange” are bordered by “I eat ” and by “” (end of line), and as a result, they can be in the same cluster. Contrary, “pple” is never bordered by “I eat” on the left and consequently cannot be put in the same cluster. This process avoid the generation of clustering links between different sub-strings of the same word.

It is important to note that the system can handle right-to-left script such as Arabic, since the left and right nodes are processing similarly.

Meta link

Meta links are used to add information about a node, such for example as *how many substitution links are related to it*. In addition, they are also used to indicate if a node can be used as a *start element*¹⁹ of an output, or which nodes can be used on the left or and the right of it to generate the cluster links. Concretely, *meta links* are used to save information needed by the output generation processing, but which are not used to evaluate the score the nodes.

¹⁹Start element can be used as first part of an output, if it is also an *end element*, it is considered as a complete sentence. Only complete sentences are outputted to the user.

3.4.2 Link characteristics

The same link characteristics are used as the previous system. However, the same links from the same nodes are packet together. Therefore, Equation (3.7) is used to calculate the *weight* of a link in function of the number of packed links. Concretely, the weight is multiplied by the number of links to retrieve the same impact on the node *score* as if there was several links.

$$W_l = weight_{base} \times n_{link} \quad (3.7)$$

- W_l is the weight of the link
- $weight_{base}$ is the base weight of the link
- n_{link} is the number of packed links

I also changed the characteristics value of each kind of link to adapt it to the new hierarchy of links. Table 3.7 shows these new values.

Table 3.7: Link characteristics

Link	Cost	Distance	Weight
Splitting	1.50	0.50	0.50
Merging	0.99	0.75	1.00
Substitution sentence	2.51	1.50	20.00
Substitution follow	0.99	1.00	7.00
Substitution part	2.51	1.50	3.50
Substitution removing	2.51	0.10	−30.00
Emotion (cf. 3.4.5)	0.99	0.25	1.00
NoEmotion	2.49	0.50	1.00

3.4.3 Forgetting process

Acquiring knowledge is a prerequisite to be able to handle various dialogues. However, more a system acquires knowledge more it needs time and computational power to visit and check all the knowledge it previously acquired. There are various methods to improve the speed of the output generation, such as use better processor or optimizing the code of the system. However, in the case of a generic dialogue agent, I think it is necessary to remove not useful knowledge to limit the size of the data to check, in a similar way to humans who forget wrong or non-meaningful information.

In order to remove some data previously acquired by the system, it is possible to use the user feedback as did in genetic algorithms. Nevertheless, an output which make the user *angry* or *not satisfy* is not necessary a *bad knowledge*. That is why I preferred a

different approach which is to remove links which are not useful for the output generation, i.e. not used during the output generation.

Concretely, links which are tested, but not followed a defined number of times (*MAXPASS*) are removed from the graph. In addition, each time the links is followed the counter is reset to zero. Figure 3.26 shows the algorithm used to check and to forget links.

```

if(link.isFollowable())
    link.count=0
else
    link.count++
if(link.count>MAXPASS)
    link.forget()

```

Figure 3.26: Algorithm used to check links which have to be forgotten

For example, the *merging links* from “a” would be removed from the graph, since the node “a” is a sub-string of many different nodes and in consequence the *cost of the link* will be too expensive to be followed (cf. Equation (3.2)), but it will be tested very frequently, and then finally be removed. In order to preserve useful links a *MAXPASS* quite big is set for the experience²⁰. However, it can be simply change to remove more or less links in function of needs.

3.4.4 Trigger value

After the system outputted a node it updates the output score *trigger value*. I changed the updating equation in order to make the trigger score higher, and the system adopt a more careful behavior. The system uses Equation (3.8) to update the trigger value. Concretely, the system simply looks for the *maximal score* of a defined number of previous nodes (set to $n = 5$ in the experiment) and multiplies it by a defined coefficient (set to $m = 2$ in the experiment). For instance, if one of the system previous outputs had a high score then the trigger value will be set to a higher value, in order to output only nodes with a maximal score.

$$V_i = \max(V_i, V_{i-1}, \dots, V_{i-n}) \times m \quad (3.8)$$

- V_i is the value of the trigger at time i
- i is the output number
- n is the number of previous output used
- m is a defined coefficient

²⁰The *MAXPASS* was set to 100.

3.4.5 Emotional processing

The system uses the six basic emotions proposed by Ekman [53]: *anger*, *disgust*, *fear*, *happiness*, *sadness*, and *surprise*. Each of them is representing by a specific tag which is recognized by the system.

These *emotional tags* can also be considered as representing emotional mimics of the user which cannot be automatically *seen* by the system, since it has no camera and no image recognition processing.

Figure 3.27 shows an example of relations between emotional nodes and usual nodes. The numbers next each link of the graph are an example of *weight* of the link, i.e. the number of relations between each node and the emotion. Explanations about the *noEmotion* tag are provided below (cf. 3.4.5).

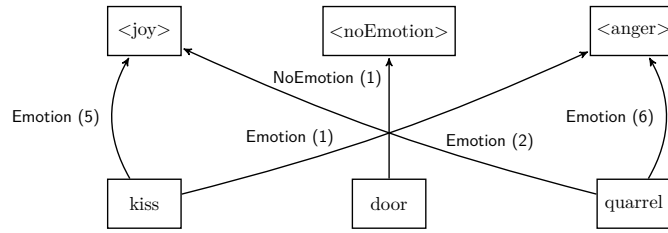


Figure 3.27: Example of graph containing emotion nodes

Concretely, the system has two kinds of behaviour, firstly it tends to have the same *emotional state* as the user, since each input emotional value influences its emotional state, and as a result the system replies *anger* input with a sentence related to this emotion. For example, if the user inputs “I am happy”, the system will reply “It’s cool”.

Second, if the output is related to an emotion it will tend to output other sentence related to the same emotion. For example, if the input is “Do you like banana” and the system response is “I hate banana”, the system will tend to output other sentences related to the same emotion (here “hate” is considered as *disgust*) such as “I don’t want it”.

Emotional state

The system uses a kind of artificial emotional heart in order to represent its current *emotional state*. This processing can be compared to other system handling emotions [73] where emotional state is used to control the output of the system. Concretely, each of the *six emotions* is represented by a value between 0 and 1. For example, a perfect *joy* state will be represented by the hash: {anger=0, disgust=0, fear=0, happiness=1, sadness=0, surprise=0, noEmotion=0}. However, in general the emotional state of the system is represented by a combination of several basic emotions.

This emotional state evolves in function of the input of the user, if he inputs sentence related to an emotion the hearth of the system will tend to the same emotion. Each atomic emotion value is calculated using Equation (3.9). Concretely, the ratio of the emotion compared to all the inputted emotions is calculated. However, I think this equation can

be changed to give more weight to the last inputted emotion, since it is supposed to be the most closely related to the current dialogue.

$$V_{emo} = \frac{N_{emo}}{N_{allEmo}} \quad (3.9)$$

- V_{emo} is the value of an emotion
- N_{emo} is the number of inputs containing the emotion
- N_{allEmo} is the number of all the emotions containing in the inputs

In order to make the system response influenced by the *sentimental value* of each node and by the current *emotional state* of the system, Equation (3.10) is used to calculate the *emotional score* of a node.

Concretely, all the basic emotions are multiplied together to get the degree of similarity between the two emotions, i.e. the system *emotional state* and the *emotion* related to the node, and then the similarity is calibrated on a defined range (from 1 to 2). For example, if the two compared emotions are the same then the node score is multiplied by two. This equation has been chosen to increase the score of nodes which match the current emotional state of the system. For example, if the user inputs “I feel bad”, then the emotional state of the system will become *bad* too, and then if we ask “How do you feel?”, the system will reply “I feel bad” instead of “I am fine”.

$$S_{emo} = S_n \times Emo_{state} \quad (3.10)$$

- S_{emo} is the score corrected with emotional state of the system
- S_n is the score of the node
- Emo_{state} is the degree of similarity (from 1 to 2) of the emotional state of the system and the emotional value of the node

NoEmotion tag

The system adds a *noEmotion tag* to each new node, then if it can relate an emotion to the node it will remove it. For example, if one of its sub-node of an *input node* is related to an emotion, the node will be related to the same emotion; potentially several emotions can be related to the same node. The *noEmotion tag* is used in order to separate nodes which have an emotion and those for which the emotion is still unknown. In addition, using this tag the system can ask the user what emotion is related to the node if no better response is available.

Figure 3.28 shows an example of dialogue where the system asks the user feeling.

U: I like peaches
S: What does it make you feel?
U: <joy>

Figure 3.28: Example of simple dialogue

In the previous example, the system had no knowledge to respond to “I like peaches”, and *no emotion* was related to it. Consequently, the system asks the user what emotion is related to this sentence in order to learn it, and then to use it in future output generation. The question used to ask emotion is provided in the language samples (cf. 3.4.6). I consider if the system acquires enough knowledge from enough users it will be able to learn the common emotion associated to each node.

Compared to the previous system, the system does not output an apology sentence such as “I cannot answer”, since even if no knowledge matches the input. The system will at least ask the user about emotion he feels.

Emotion link

Emotion links are set between a knowledge node and a basic emotion. Several emotion links can be set to a same node. Those links can be replaced by simple *substitution link*. However, since emotion computation is an important part of this research, I chose to use a specific link to be able to set its parameters independently.

I also created a specific link for the *noEmotion* tag, the *noEmotion link*, in order to reduce its influence on the output generation and to avoid asking the user emotion if other outputs were available.

3.4.6 Starting knowledge

The training samples are the same as the previous research, except some new samples containing *emotional tags*. These new samples was manually created, four samples for the *joy* emotion and two for the other emotions has been created. This is because I consider that the *joy* emotion is the most often used emotion in casual dialogues.

Figure 3.29 shows an example of emotional sample. The objective of those samples is to teach the system emotions related to some simple emotionally strongly related nodes.

```

U1: I love
U2: <joy>
<init>
U1: <noEmotion>
U2: What does it make you fell?
<init>
U1: <anger>
U2: Keep calm

```

Figure 3.29: Example of emotion sample

In the example, the node “I love” and all its sub-nodes such as “I” or “love”, will be related to the <joy> node.

Blacklisted sub-strings

At the exception of Chinese for which the system does not use *splitting control*, the system used three manually selected strings for each language. For English they were: “app”, “eak” and “pe”.

3.4.7 Experiments

In order to carry out comparison experiment I kept the same process as the previous system.

In order to let the users able to input any of the six emotions, six buttons have been added to the *Android application* interface. The subjects used them freely; particularly to answer emotion questions of the system.

Experiments settings

Table 3.8 shows some information about the samples taught to the system and the resulting generated graph. The same methods as **3.3.10** are used to count words.

Table 3.8: Corpus information

Number	Zh	En	Fr	Ja	Ko
of nodes	1623	1283	1216	2038	1687
of words	1276	1331	1625	1552	1332
of characters	1873	5949	8191	3464	3480
of sentences (knowledge)	100	100	100	100	100
of sentences (dialogues)	212	209	277	255	248

Table 3.9 shows the information relative to the subject participating to the experiment.

Table 3.9: Subject’s information

	Zh	En	Fr	Ja	Ko	Total
Subject	5	4	5	14	5	33
Male	2	3	5	5	0	15
Female	3	1	0	9	5	18
Student	5	3	4	9	5	26
Worker	0	1	1	5	0	7
Age [years]	21.8	25.0	25.8	20.6	21.0	22.2

Experiments results

Figure 3.30 shows a part of a conversation between the system and a subject of the experiment with the evaluation of the response of the system in parenthesis.

U: How are you?
S: I am fine (*correct*)
S: Nice to meet you (*semi-correct*)
S: Pretty good (*correct*)
U: Today I’m tired
S: What does it make you feel? (*semi-correct*)
U: <sadness>
S: Don’t worry (*correct*)
U: I like video games
S: What does it make you feel? (*correct*)
U: <joy>
S: Oh, it’s cool (*correct*)

Figure 3.30: Example of a dialogue with the system

According to the subject, the responses “Nice to meet you” and “What does it make you feel?” were *semi-correct* since they were in relation with the input, but not natural.

Table 3.10 shows the precision of the speech recognition input method for each language of the experiment.

Table 3.10: Speech recognition precision

	Zh [%]	En [%]	Fr [%]	Ja [%]	Ko [%]	Mean [%]
Precision	90.4	81.9	95.8	91.1	88.7	89.6

Table 3.11 shows the percentage of inputs which were emotions for each language.

Table 3.11: Number of emotion

	Zh [%]	En [%]	Fr [%]	Ja [%]	Ko [%]	Mean [%]
Emotion	14.3	13.1	13.3	16.1	8.7	13.1

Table 3.12 contains the percentage of each emotion input by the user in each different language. The *joy* emotion was the most often input. I consider it is the most easily identifiable emotion and the users have no embarrassment to use it.

Table 3.12: Input emotion

	Zh [%]	En [%]	Fr [%]	Ja [%]	Ko [%]	Mean [%]
Anger	6.9	8.0	3.2	4.0	0.0	4.4
Disgust	10.3	8.0	16.1	5.9	0.0	8.1
Fear	6.9	8.0	0.0	4.9	0.0	4.0
Joy	27.6	56.0	51.6	59.5	89.5	54.8
Sadness	13.8	8.0	25.8	11.9	10.5	14.0
Surprise	34.5	12.0	3.2	23.8	0.0	14.7

Table 3.13 shows the results of the experiment of the proposed system and of the two baselines.

Results analysis

First, the results of the proposed system exceed the two multilingual baselines and are similar (English) or exceed the result of a language specific ELIZA system.

We can note that results do not depend of type of characters used by the language, languages which use Latin characters obtained similar results than those which use a different script.

However, I consider that since the system does not give a special meaning to the spaces present in the script as a human naturally does, the *splitting process* is more complicated in language using space. In fact, I consider that spaces are used in those languages to

Table 3.13: Experiments results

Language	Correct [%]	Semi-correct [%]	Erroneous [%]	Precision [%]
Chinese	33.0	19.7	47.3	52.7
English	41.4	19.9	38.7	61.3
French	39.9	15.4	44.6	55.4
Japanese	40.7	14.7	44.4	55.6
Korean	32.9	18.9	48.2	51.8
Mean	37.6	17.8	44.7	55.3
Baseline 2				
Chinese	25.9	17.0	57.1	42.9
English	39.1	14.2	46.7	53.3
French	33.9	9.9	56.1	43.9
Japanese	31.7	16.2	52.1	47.9
Korean	34.4	23.1	42.6	57.5
Mean	33.0	16.1	50.9	49.1
Baseline 1.1 (Baseline 1.2)				
Chinese	25.6 (16.0)	13.6 (30.4)	4.4 (10.0)	39.2 (46.4)
English	4.0 (8.4)	16.0 (53.2)	15.2 (3.2)	20.0 (61.6)
German**	17.6 (19.2)	19.6 (20.0)	13.6 (10.0)	37.2 (39.2)
Japanese	14.6 (38.2)	2.5 (13.5)	8.9 (22.3)	17.1 (51.7)
Mean	15.4 (20.4)	12.9 (29.3)	10.5 (11.4)	28.3 (49.7)

Baseline 1.1: Spoken Dialogue Method Using Inductive Learning with Genetic Algorithm (GA-ILSD)

Baseline 1.2: ELIZA-like system

Baseline 2: System without emotions

** Baseline 1's experiment has also been carried out in German.

fix the problem of the difficulty to recognize each word. The system cannot graphically analyses characters to classify them, but if it acquires enough knowledge it can solve this problem in a similar way as *scriptio continua* where no space was used, but human reader was able to understand it.

Impact of the emotion processing The emotion processing permits the system to answer to some inputs using only their emotion value. Second, asking the subject for emotion gives a better impression than simply outputting an apology sentence. I consider that is one reason why the system exceeds the previous system which was very similar on other part of the output generation.

In addition, since only few emotional relations was set to the system at starting of the dialogue, all the relations taught by the user have an important influence on the system response to emotion input. For example, if the user inputs the *joy tag* after the system

output the node “I bought a car”, this will be related to the *joy emotion*, and if the user inputs “I crash my car” the system may reply “It’s cool”, since the node “car” is in the user input and is included in the *joy* related node “I bought a car”. However, if the system learns that the node “crash” is not *joyful*, it will be able to avoid incorrect outputs.

Difference between languages The results are similar in all the different languages. However, I consider that each language specificity’s have an influence on the comportment of the system. For example, in language such as Korean and Japanese the politeness of the sentence is mostly represented by a verb inflection, in consequence the different levels of politeness are very similar, and as a result the system tends to select all of them as output even if they have the same meaning. The same problem occurs in other languages, however, since different levels of politeness use different constructions, the system uses the most similar construction, and then replies with the same register.

In order solve this problem, the system must be able to select the correct politeness level in function of the context. I consider this can be done using the same algorithm, but with a very fine granularity in emotion and context input. For example, the age of the user could be input in the system with each input, for example, <“Hi”, <60 years old>> would be replied with “Hello, sir”, but if the input is <“Hi”, <15 years old>> the system will output “Hi, guy”.

3.4.8 Conclusion

I proved that a *language-independent dialogue system* can be easily adapted to provide a *multimodal dialogue system*, since non-verbal inputs can be handled similarly to verbal input. As a consequence, I consider that any other kinds of input could be added to the system just like it has been done for the emotions.

In addition, adding a simple emotion processing improved the system results. Even if the results are not better than the state of art of dialogue agents, the system can be easily improved in order to approach the result of language-dependent system, especially by teaching more knowledge to the system.

As a result, I consider handle many different kinds of input at the same time, such as time, user movement, weather and so on, will increase the system until it can exceed more static systems’ results.

3.5 Conclusion

In order to achieve the final objective of this research which is to create a dialogue agent which can have a casual conversation in real-time as well as be able to reply correctly some questions. I developed a fully *language-independent algorithm*. Using simple relations between nodes, the system can represent all the knowledge required to generate an output to the user input. In addition, in order to make the system as flexible as possible the nodes do not represent only sentences words, but also parts of sentence or part of word. This approach is different from a rule-based or grammar-based method, however it offers

much more flexibility and I consider it as more close to the method used by human to create a response. Especially, humans do not need to know grammar to speak a language correctly, many native speakers of a language speak it instinctively.

In addition, the used graph can be upgraded to save states, for example, it can be used to save the contextual state or the system internal state in order to check the consistency of outputs. The proposed method is already able to save some simple context information and use them to influence the output generation. However, the research must be continued to implement more precise consistency checking. Nevertheless, I consider the system base does not have to be changed to implement it.

On first hand, language-independent algorithms are not necessary to create a dialogues agent. However, since the language used by each user is unique and new terms are continuously incorporated in a language it is necessary to create a system which can adapt to any evolution or particularity of the user language. As a consequence, this kind of systems is intrinsically language-independent and I consider the proposed method is a good base for further language-independent systems which will attract more attention in future.

3.6 Future works

The previously experimented system obtained satisfying results, however it needs to be improved as well in performance as in precision to be able to carry out more natural and complex dialogues. Therefore, I thought to several ways of evolution for the systems.

Behaviour The proposed system looks for all the sentences which match the input, as a result it often outputs many similar sentences, such as “I ate”, then “I ate breakfast”, and then “Today, I ate breakfast”. I think it is possible to avoid generating similar responses using *negative links*. Concretely each link can be used in an opposite way to decrease the score of nodes closed from the previously outputted nodes, like did with the *substitution removing link*.

Moreover, in order to improve the quality of the dialogue, I consider that *link characteristics* could be adjusted to make the system firstly answer the question, and then secondly ask questions related to the same subject. However, since setting an adequate value to each link requires a lot of time, this task would be better conducted automatically using a learning algorithm such as reinforcement learning [74], which is easy to implement in a dialogue system using the user feedback to evaluate the system outputs and the used links.

In addition, it would also be interested to allow the user to ask the system current feeling, or its feeling about something, for example using an input such as “How do you feel now?”. This behaviour could be easily implemented using the system current *emotional state*.

Emotion input For the experiment buttons were used to input emotion, however, in order to acquire emotion automatically from the user it is also possible to use image pro-

cessing to analyze emotions provided by the user mimics or use a *EEG emotion judgment system* [75].

Context processing The system can be set up to keep *active nodes* longer in order to use them as *context*. Concretely, the score of previously used nodes can be kept upper zero to use them in priority for new output generation. For example, if the user inputs “I like fruits” and then “How about you?”, the system will use nodes related to previous input in priority and, for example, output “I like vegetables” rather than “I am fine”.

Knowledge sharing In order to acquire more knowledge, sharing it between several users like studied in 2.4 would help the system to acquire many different kinds of knowledge directly from the users in order to facilitate the language acquisition task.

Learning from examples In order to identify words’ categories, it is possible to using example words such as “cats” or “fruits” and to look for similar words, which are words which have a similar number of a defined kind of links. Concretely, the system can use a list of previously defined words and use them as a reference to count the number of links associated to them similarly as did for the *splitting control*. For example, using the nodes “apple”, “human” and “dog”, the system can calculate the number of splitting links related to a generic noun, and then use it during the clustering process. However, the list of these samples words is language-dependent. Nevertheless, they can be considered as specific words taught to the system, similarly to example words provided by school teachers to students.

Simultaneous multilingual system In order to check that the system is really multilingual, instead of checking each language separately, it is possible to teach all the knowledge in all the target languages at the same time. As a consequence, the user could input sentence in several different languages during the same dialogue. For example, it would be possible to start a conversation in Japanese, continue it in Chinese and finish it in English. Since the emotional context is the same in all the language it could be used normally. However, the other context parts could not be used easily after a *language switching*. In order to keep, context the system need to acquire links between nodes of different languages. Therefore, I think links acquired during a multilingual conversation can be used to relate words which are semantically related in different languages.

Chapter 4

Conclusion

4.1 Conclusion

The field of dialogue agents is wild and includes many different kinds of system. During my research, I aimed to create generic systems which can be a basis to other kinds of more specific system. Firstly, I focused on one language; Japanese, then I enlarged the aim of the research to language-independent systems. Focusing on one language helps to simplify the development of dialogue agents. However, more generic dialogue and high precision are aimed less external tools can be used, and as result starting from one language is not considered that beneficial. I consider any dialogue agent will tend to be language-independent in order to improve its learning faculty and flexibility. As a consequence, in a second hand, I chose to develop a new method for dialogue agent focusing on language-independent from the beginning of the development of the system.

First, the proposed *language-specific system* is able to handle a simple conversation and to acquire knowledge from the user. In addition, it can be easily adapted to change the manner it shares knowledge. Moreover, I checked that speaking in a human way is not consistently the best method to provide information to the user. In function of the environment where the system is used, other manners of speaking can be more adequate. Nevertheless, I consider understanding the user's natural speaking is the most important point of a dialogue system. After understanding the user input, the system can proceed to a correct output generation which can be easily adapted in function of the aimed behavior.

Second, the proposed *language-independent system* is flexible and can be set up to handle any kind of input and is considered to be suited for developing *multimodal dialogue agents*. The precision of the proposed system is still relatively low, however the system can be improved especially by providing more knowledge and by increasing the number of non-verbal inputs. In addition, the system is real-time and can accept output at any time. I consider accepting real-time input is an important point to make a dialogue system more lively and user-friendly. Concretely, the proposed language-independent method has many capabilities and can be upgraded easily in order to improve the system.

Both of the proposed systems have advantages and drawbacks. *Language-dependent* systems provide relatively good results very quickly even with a limited knowledge and

can be improved simply by adding them more knowledge. However, they are hard to make lively since they use a very static output generation and cannot handle any kind of input like humans who are intrinsically multimodal. On another hand, *language-independent* systems require more knowledge to learn properly the target language, but their behaviour is much more flexible and they can learn any kind of input like humans do. Conclusively, I consider the next step of dialogue agent is to combine the advantages of the two systems to create a robust flexible language-independent system which can acquire knowledge from the users and from any kind language-specific resources.

Moreover, future dialogue agents must be able to check the consistency of their outputs with precision in order to avoid incorrect and unnatural behaviors. The consistency cannot be only checked using textual input but needs other kind of environmental information. For example, time is necessary to check if the output “Good morning” is correct or not. That is why in this context I consider language-independent dialogue agents are more well-disposed than language-specific ones.

In this research I showed that it is possible to develop a system based on one unique main algorithm which can be used to handle several kinds of input. A same model can be used to represent all the knowledge acquired by the system and all the links between them. That is why I think using external tools and resources as simple inputs and not in the core processing would help to develop flexible dialogue agents. However, external tools and resources can be used to inject knowledge in the system. In addition, working on several languages gives useful advice to check the validity of the method and to improve it, since it emphasizes the impact of each processing on different grammar structures.

4.2 Future works

Creating a dialogue agent which can handle conversation comparable to human¹ is complicated and requires the combination of many methods. That is why I consider it still requires many works. However, using a generic algorithm can help to achieve this goal. In a similar way as the *reduced instruction set computer (RISC)*², a set of simple processing can be used to carry out the same work as more high level instructions and if they are executed enough efficiently the result can be as reliable as more complex processing.

In order to increase the language-independent proposed system, firstly optimizations are needed. The human brain contains about 100 billion neurons [76] and 100 to 500 trillion of synapses [77]. Even if a dialogue agent does not need the same number of elements and links, I consider a same order of magnitude is needed. In order to hold this huge number of objects with current computers, the dialogue agent must be optimized as possible to be able to handle knowledge as much as possible.

In addition, a language-independent algorithm can be used to handle any kind of input and not only text or verbal input. For example, it can also handle gestures, emotions or movements. In order to make a human-like system non-verbal inputs are very important since they provide an important part of the information transferred during a conversation.

¹For example which can accomplish the *Turing test*.

²By opposition to the *complex instruction set computer (CISC)*.

Moreover, the non-verbal inputs include physical information such as gestures, but they can also include tasks such as *<sending a mail>*. For instance, if the input is the sentence “Send the mail to my mother”, the system has to output the sentence “I send a mail to your mother” and a non-verbal output, the task *<sending a mail to user mother>*. This approach would be a solution to merge *task-oriented* and *non-task-oriented systems* into a unique generic system.

That is why I consider new dialogue agents must be as generic as possible in order to be able to handle many different kinds of input. These systems will be *multimodal* and are in a better position to reach the user expectations, namely providing *casual chatting* as well as accomplishing *specific tasks*. In addition all the information acquired by the system can be used to check the consistency of outputs and avoid contradictions. For example, the *joy* emotion is not coherent with a *sad face*; therefore, using several kinds of information a dialogue agent can find these kinds of contradiction and looks cleverer. Concretely, I consider future dialogue agents will be *language-independent multimodal* and *consistent*.

Moreover, as I checked in the first part of my research, a dialogue agent does not have to speak like a human in all the situations, but needs to be adapted to them. That is why I think language-independent agent behavior could also be adapted to the situation where they are used. Concretely, this behavior can be defined directly from the samples used to train it, i.e. the system acquires and replicates the behavior of the dialogue samples.

Bibliography

- [1] J. Weizenbaum, “ELIZA - a computer program for the study of natural language communication between man and machine,” *Communications of the ACM* 9 (1), vol. 36-45, 1966.
- [2] B. Batacharia, D. Levy, R. Catizone, A. Krotov, and Y. Wilks, “Converse: a conversational companion,” in *Machine conversations*, pp. 205–215, Springer, 1999.
- [3] J. Jia, “CSIEC: A computer assisted english learning chatbot based on textual knowledge and reasoning,” *Knowledge-Based Systems*, vol. 22, no. 4, pp. 249–255, 2009.
- [4] S. Higuchi, R. Rzepka, and K. Araki, “A casual conversation system using modality and word associations retrieved from the web,” in *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pp. 382–390, Association for Computational Linguistics, 2008.
- [5] D. Ferrucci, E. Brown, J. Chu-Carroll, J. Fan, D. Gondek, A. A. Kalyanpur, A. Lally, J. W. Murdock, E. Nyberg, J. Prager, *et al.*, “Building watson: An overview of the deepqa project,” *AI magazine*, vol. 31, no. 3, pp. 59–79, 2010.
- [6] J. Edlund, J. Gustafson, M. Heldner, and A. Hjalmarsson, “Towards human-like spoken dialogue systems,” *Speech Communication*, vol. 50, no. 8–9, pp. 630 – 645, 2008. Evaluating new methods and models for advanced speech-based interactive systems.
- [7] A. M. Turing, “Computing machinery and intelligence,” *Mind*, pp. 433–460, 1950.
- [8] A. Jordan and K. Araki, “Spoken dialog processing for acquiring taste and knowledge,” *Proceedings of PACLING2013*, 2013.
- [9] A. Jordan and K. Araki, “Real-time language-independent algorithm for dialogue agents,” *Journal of Japan Society for Fuzzy Theory and Intelligent Informatics*. in press.
- [10] T. Winograd, “Understanding natural language,” *Cognitive psychology*, vol. 3, no. 1, pp. 1–191, 1972.
- [11] R. Wallace, “The elements of AIML style,” *Alice AI Foundation*, 2003.

- [12] R. S. Wallace, “The anatomy of A.L.I.C.E.,” *Parsing the Turing Test*, pp. 181–210, 2009.
- [13] S. Ontañón and E. Plaza, “Multiagent inductive learning: an argumentation-based approach,” in *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, pp. 839–846, 2010.
- [14] K. Araki and M. Kuroda, “Generality of spoken dialogue system using SeGa-IL for different languages,” in *Proceeding of the IASTED International Conference COMPUTATIONAL INTELLIGENCE*, pp. 70–75, 2006.
- [15] W. Banzhaf, P. Nordin, R. E. Keller, and F. D. Francone, *Genetic programming: an introduction*, vol. 1. Morgan Kaufmann San Francisco, 1998.
- [16] C. Hori, K. Ohtake, T. Misu, H. Kashioka, and S. Nakamura, “Statistical dialog management applied to wfst-based dialog systems,” in *Speech and Signal Processing, ICASSP 2009. IEEE International Conference on Acoustics*, pp. 4793–4796, IEEE, 2009.
- [17] J. Bang, H. Noh, Y. Kim, and G. G. Lee, “Example-based chat-oriented dialogue system with personalized long-term memory,” in *International Conference on Big Data and Smart Computing (BigComp)*, pp. 238–243, IEEE, 2015.
- [18] H. Wang, Z. Lu, H. Li, and E. Chen, “A dataset for research on short-text conversations,” in *EMNLP*, pp. 935–945, 2013.
- [19] Kurohashi & Kawahara lab., “Japanese morphological analysis system JUMAN version 7,” 2012.
- [20] T. Kudo and Y. Matsumoto, “Fast methods for kernel-based text analysis,” in *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics*, vol. 1, pp. 24–31, Association for Computational Linguistics, 2003.
- [21] G. A. Miller, “Wordnet: A lexical database for english,” *Commun. ACM*, vol. 38, pp. 39–41, Nov. 1995.
- [22] C. F. Baker, C. J. Fillmore, and J. B. Lowe, “The berkeley framenet project,” in *Proceedings of the 17th international conference on Computational linguistics*, vol. 1, pp. 86–90, Association for Computational Linguistics, 1998.
- [23] J. Curtis, G. Matthews, and D. Baxter, “On the effective use of cyc in a question answering system,” in *In Proceedings of the IJCAI Workshop on Knowledge and Reasoning for Answering Questions*, pp. 61–70, 2005.
- [24] H. Liu and P. Singh, “Conceptnet-a practical commonsense reasoning tool-kit,” *BT technology journal*, vol. 22, no. 4, pp. 211–226, 2004.
- [25] C. D. Manning and H. Schütze, *Foundations of statistical natural language processing*. MIT press, 1999.

- [26] H. Dewar, P. Bratley, and J. P. Thorne, “A program for the syntactic analysis of english sentences,” *Commun. ACM*, vol. 12, pp. 476–479, Aug. 1969.
- [27] D. Kawahara and S. Kurohashi, “A fully-lexicalized probabilistic model for japanese syntactic and case structure analysis,” in *Proceedings of the main conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*, pp. 176–183, Association for Computational Linguistics, 2006.
- [28] M. R. Quillan, “Semantic memory,” tech. rep., DTIC Document, 1966.
- [29] M. Minsky, “A framework for representing knowledge,” tech. rep., Massachusetts Institute of Technology, Cambridge, MA, USA, 1974.
- [30] A. Jordan and K. Araki, “Iden-teki arugorizumu o mochiita kinōteki gakushū ni yoru onsei taiwa shori shisutemu GA-ILSD ni okeru shitsumon ōtō nōryoku ni tsuite,” in *Language Acquisition and Understanding*, vol. 7, 2012. in Japanese.
- [31] S. Kita and S. Ide, “Nodding, aizuchi, and final particles in japanese conversation: How conversation reflects the ideology of communication and social relationships,” *Journal of Pragmatics*, vol. 39, no. 7, pp. 1242 – 1254, 2007. Nodding, Aizuchi, and Final Particles in Japanese Conversation.
- [32] M. Takeuchi, N. Kitaoka, and S. Nakagawa, “A spoken dialog system activating the natural response timing using prosodic and linguistic information for chat-like conversation,” *IPSJ SIG Technical Report*, 2004.
- [33] T. Misu, T. Kawahara, T. Shoji, and M. Minoh, “Speech-based interactive information guidance system using question-answering and information recommendation,” *Journal of Information Processing*, 2007.
- [34] D. R. HEISE, “The semantic differential and attitude research,” *Attitude Measurement*, pp. 235–253, 1970.
- [35] K. Anzou, K. Murata, and M. Numasaki, *Shakai Shinrigaku Kenkyuu Nyuumon*. University of Tokyo Press, September 2009.
- [36] Y. Uchida and K. Araki, “Evaluation of a System Noun Concepts Acquisition from Utterances about Images (SINCA) Using Daily Conversation Data,” *Proceedings of NAACL-HLT*, pp. 65–68, 2009.
- [37] S. Varges, F. Weng, and H. Pon-Barry, “Interactive question answering and constraint relaxation in spoken dialogue systems,” *Natural Language Engineering*, vol. 15, no. 01, pp. 9–30, 2009.
- [38] J. Gustafson, A. Larsson, R. Carlson, and K. Hellman, “How do system questions influence lexical choices in user answers,” in *In Proc. Eurospeech ’ 97*, pp. 2275–2278, 1997.

- [39] Oracle, “Jdk 6 swing (java foundation classes (jfc))-related apis and developer guides,” 2013. [Online; accessed 21-September-2013].
- [40] R. Jiang, R. E. Banchs, S. Kim, K. H. Yeo, A. Niswar, and H. Li, “Web-based multimodal multidomain spoken dialogue system,” in *Proceedings of 5th International Workshop on Spoken Dialog Systems*, 2014.
- [41] M. Ueno, N. Mori, and K. Matsumoto, “Novel chatterbot system utilizing web information,” in *Distributed Computing and Artificial Intelligence*, pp. 605–612, Springer, 2010.
- [42] Y. Uchida and K. Araki, “Evaluation of a system for noun concepts acquisition from utterances about images (SINCA),” *Proceedings of the Fuzzy System Symposium*, vol. 24, pp. 72–72, 2008.
- [43] B. Golosio, A. Cangelosi, O. Gamotina, and G. L. Masala, “A cognitive neural architecture able to learn and communicate through natural language,” *PLoS ONE*, vol. 10, 11 2015.
- [44] R. T. McDonald, J. Nivre, Y. Quirnbach-Brundage, Y. Goldberg, D. Das, K. Ganchev, K. B. Hall, S. Petrov, H. Zhang, O. Täckström, *et al.*, “Universal dependency annotation for multilingual parsing,” in *ACL (2)*, pp. 92–97, 2013.
- [45] S. Buchholz and E. Marsi, “Conll-x shared task on multilingual dependency parsing,” in *Proceedings of the Tenth Conference on Computational Natural Language Learning*, pp. 149–164, Association for Computational Linguistics, 2006.
- [46] Y. Kim, “Convolutional neural networks for sentence classification,” *arXiv preprint arXiv:1408.5882*, 2014.
- [47] L. R. Rabiner, “A tutorial on hidden markov models and selected applications in speech recognition,” *Proceedings of the IEEE*, vol. 77, no. 2, pp. 257–286, 1989.
- [48] A. Ritter, C. Cherry, and W. B. Dolan, “Data-driven response generation in social media,” in *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pp. 583–593, Association for Computational Linguistics, 2011.
- [49] F. Bessho, T. Harada, and Y. Kuniyoshi, “Dialog system using real-time crowdsourcing and twitter large-scale corpus,” in *Proceedings of the 13th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pp. 227–231, Association for Computational Linguistics, 2012.
- [50] S. Brin and L. Page, “The anatomy of a large-scale hypertextual web search engine,” in *Seventh International World-Wide Web Conference (WWW 1998)*, 1998.
- [51] A. Jordan and K. Araki, “A framework for multilingual real-time spoken dialogue agents,” *Proceedings of iCAST2014*, 2014.

- [52] H. C. Warren, *A history of the association psychology*. Charles Scribner’s Sons, 1921.
- [53] P. Ekman, “Universals and cultural differences in facial expression of emotion,” *J. Cole ed. Nebraska Symposium on Motivation.*, pp. 207–283, 1972.
- [54] B. Dorow and D. Widdows, “Discovering corpus-specific word senses,” in *Proceedings of the tenth conference on European chapter of the Association for Computational Linguistics*, vol. 2, pp. 79–82, Association for Computational Linguistics, 2003.
- [55] D. Hope and B. Keller, “Maxmax: a graph-based soft clustering algorithm applied to word sense induction,” in *Computational Linguistics and Intelligent Text Processing*, pp. 368–381, Springer, 2013.
- [56] D. Hope and B. Keller, “UoS: A graph-based system for graded word sense induction,” *Atlanta, Georgia, USA*, p. 689, 2013.
- [57] A. Jain, D. K. Tayal, and S. Yadav, “Retrieving web search results using max–max soft clustering for hindi query,” *International Journal of System Assurance Engineering and Management*, pp. 1–12, 2014.
- [58] Y. Matsumoto, A. Kitauchi, T. Yamashita, Y. Hirano, H. Matsuda, K. Takaoka, and M. Asahara, “Morphological analysis system chasen version 2.2. 9 manual,” *Nara Institute of Science and Technology*, 2002.
- [59] D. Mochihashi, T. Yamada, and N. Ueda, “Bayesian unsupervised word segmentation with hierarchical language modeling,” *IPSJ SIG Notes*, vol. 2009, p. 49, mar 2009.
- [60] L. Gammaitoni, P. Hänggi, P. Jung, and F. Marchesoni, “Stochastic resonance,” *Reviews of modern physics*, vol. 70, no. 1, p. 223, 1998.
- [61] G. Chowdhury, *Introduction to modern information retrieval*. Facet publishing, 2010.
- [62] G. Lakoff and M. Johnson, “The metaphorical structure of the human conceptual system,” *Cognitive science*, vol. 4, no. 2, pp. 195–208, 1980.
- [63] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A.-r. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath, *et al.*, “Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups,” *Signal Processing Magazine, IEEE*, vol. 29, no. 6, pp. 82–97, 2012.
- [64] M. Kuroda and K. Araki, “Evaluation of generality of sega-ilsd for a chat using different languages,” *IPSJ SIG Notes*, vol. 2007, pp. 79–85, jan 2007.
- [65] S. Sekine and R. Grishman, “A corpus-based probabilistic grammar with only two non-terminals,” in *proceedings of the International Workshop on Parsing Technologies*, vol. 95, 1995.

- [66] H.-P. Zhang, H.-K. Yu, D.-Y. Xiong, and Q. Liu, “HHMM-based chinese lexical analyzer ictclas,” in *Proceedings of the second SIGHAN workshop on Chinese language processing*, vol. 17, pp. 184–187, Association for Computational Linguistics, 2003.
- [67] S. Park, D. Choi, E.-k. Kim, and K.-S. Choi, “A plug-in component-based korean morphological analyzer,” in *Proceedings of HCLT 2010*, pp. 197–201, 2010.
- [68] T. Menzies, A. Dekhtyar, J. Distefano, and J. Greenwald, “Problems with precision: A response to ’comments on ’data mining static code attributes to learn defect predictors’,” *IEEE Transactions on Software Engineering*, vol. 33, no. 9, p. 637, 2007.
- [69] M. Ptaszynski, P. Dybala, R. Rzepka, and K. Araki, “Effective analysis of emotiveness in utterances based on features of lexical and non-lexical layers of speech,” *Proceedings of NLP*, pp. 171–174, 2008.
- [70] D. J. Litman and K. Forbes-Riley, “Predicting student emotions in computer-human tutoring dialogues,” in *Proceedings of the 42Nd Annual Meeting on Association for Computational Linguistics*, ACL ’04, (Stroudsburg, PA, USA), Association for Computational Linguistics, 2004.
- [71] J. Pittermann, A. Pittermann, and W. Minker, “Emotion recognition and adaptation in spoken dialogue systems,” *International Journal of Speech Technology*, vol. 13, no. 1, pp. 49–60, 2010.
- [72] T. Hasegawa, N. Kaji, N. Yoshinaga, and M. Toyoda, “Predicting and eliciting addressee’s emotion in online dialogue,” in *ACL (1)*, pp. 964–972, 2013.
- [73] S. Larsson and D. R. Traum, “Information state and dialogue management in the trindi dialogue move engine toolkit,” *Natural language engineering*, vol. 6, no. 3&4, pp. 323–340, 2000.
- [74] L. P. Kaelbling, M. L. Littman, and A. W. Moore, “Reinforcement learning: A survey,” *Journal of artificial intelligence research*, pp. 237–285, 1996.
- [75] M. Morimoto, M. Imono, S. Tsuchiya, and H. Watanabe, “Construction of the eeg emotion judgment system using concept base of eeg features,” in *Proceedings of ICAI2015*, pp. 485–490, 2015.
- [76] R. W. Williams and K. Herrup, “The control of neuron number,” *Annual review of neuroscience*, vol. 11, no. 1, pp. 423–453, 1988.
- [77] D. A. Drachman, “Do we have brain to spare?,” *Neurology*, vol. 64, no. 12, pp. 2004–2005, 2005.
- [78] A. Jordan and K. Araki, “Evaluation of a standalone language-independent dialogue framework,” in *Proceedings of ICAI2015*, pp. 316–322, 2015.
- [79] A. Jordan and K. Araki, “Comparison of two knowledge treatments for questions answering,” *Proceedings of SNLP2013*, 2013. Best Student Paper Award.

Acknowledgements

I would like to express deep appreciation to the associate examiners Professor Tsuyoshi Yamamoto (Laboratory of Information Media Environment, Graduate School of Science and Technology, Hokkaido University), Professor Miki Haseyama (Laboratory of Media Dynamics, Graduate School of Science and Technology, Hokkaido University) and Professor Yuji Sakamoto (Laboratory of Media Creation Methodology, Graduate School of Science and Technology).

I also wish to express great appreciation to my doctoral course academic adviser Professor Kenji Araki (Language Media Laboratory, Graduate School of Science and Technology, Hokkaido University) for his guidance and support in the field of natural language processing.

I am also grateful to Assistant Professor Rafal Rzepka for his advice and ideas he has given to me over the course of this research, in particular concerning my last journal paper.

I would also like to thank all my fellow students who helped me in various ways.

Research achievements

Journal Papers

- A. Jordan and K. Araki, “Real-time language-independent algorithm for dialogue agents,” *Journal of Japan Society for Fuzzy Theory and Intelligent Informatics*. in press

International Conferences

- A. Jordan and K. Araki, “Evaluation of a standalone language-independent dialogue framework,” in *Proceedings of ICAI2015*, pp. 316–322, 2015
- A. Jordan and K. Araki, “A framework for multilingual real-time spoken dialogue agents,” *Proceedings of iCAST2014*, 2014
- A. Jordan and K. Araki, “Comparison of two knowledge treatments for questions answering,” *Proceedings of SNLP2013*, 2013. Best Student Paper Award
- A. Jordan and K. Araki, “Spoken dialog processing for acquiring taste and knowledge,” *Proceedings of PACLING2013*, 2013

National Conferences and Meetings

- ジョルダン・アルノ, “様々な行動を扱える対話システムの枠組みに関する研究,” 年北大情報系若手連携シンポジウム, pp. 117–322, 11 月 28 日-29 日 2013 年. ポスター発表.
- ジョルダン・アルノ, 荒木健治, “遺伝的アルゴリズムを用いた帰納的学習による音声対話処理システム GA-ILSD における質問応答能力について,” 言語獲得と理解研究会報告第 7 号, 2012 年.