Universidade de Aveiro
2017

Departamento de Electrónica,
Telecomunicações e Informática
Programa de Doutoramento em Informática
das Universidades de Aveiro, Minho e Porto

José Casimiro Pereira

Geração de Linguagem Natural no Âmbito de Interação Multimodal em Português:
Conversão de dados para texto baseada em tradução automática

Natural Language Generation in the Context of Multimodal Interaction in Portuguese:
Data-to-Text based in automatic translation

Universidade de Aveiro
2017

Departamento de Electrónica,
Telecomunicações e Informática
Programa de Doutoramento em Informática
das Universidades de Aveiro, Minho e Porto

José Casimiro Pereira

Geração de Linguagem Natural no Âmbito de Interação Multimodal em Português:
Conversão de dados para texto baseada em tradução automática

Natural Language Generation in the Context of Multimodal Interaction in Portuguese:
Data-to-Text based in automatic translation

Tese apresentada à Universidade de Aveiro, Minho e Porto para cumprimento dos requisitos necessários à obtenção do grau de Doutor em Informática no âmbito do Programa Doutoral MAP-i, realizado sob a orientação científica do Doutor António Joaquim da Silva Teixeira, Professor Associado do Departamento Electrónica Telecomunicações e Informática da Universidade de Aveiro, e pelo Professor Doutor Joaquim Manuel Henriques de Sousa Pinto, Professor Auxiliar do Departamento Electrónica Telecomunicações e Informática da Universidade de Aveiro.

Agradecimentos

Terminado este percurso, iniciado em 1994, quando entrei na licenciatura em Matemática Aplicada e Computação, aqui na UA, o meu sentimento é de profunda satisfação. Terminado este longo período, não posso deixar de agradecer às pessoas que, desde sempre, me acompanharam, incentivaram e suportaram todas as minhas ausências e acessos de mau humor.

Em primeiro lugar, uma palavra de ENORME reconhecimento à minha família, nas pessoas da minha esposa, Maria dos Anjos, dos meus filhos, Beatriz, Isabel e Edgar, e dos meus pais e sogros.

Aos meus orientadores. Ao Professor Joaquim Sousa Pinto por ter acreditado em mim quando fiz o meu mestrado. Ao Professor António Teixeira, sem o qual, nunca teria conseguido terminar este doutoramento. Aos dois, pela grande paciência e empenhamento, o meu muito obrigado.

Aos meus colegas no IPT, que sempre com grande sacrifício, tudo fizeram para que a minha carga letiva fosse a menor possível, para poder ter tempo para me dedicar a este projeto.

Ao meus amigos, pelas enormes ausências...

A todos os que de uma forma, ou de outra, apesar de aqui não enumerados, contribuiram para a realização deste trabalho. A todos, o meu Bem Hajam.

Palavras-Chave conversão de dados para texto, geração de linguagem natural, tradução automática, tradução baseada em sintagmas, avaliação, Português

Resumo

Para ser possível a interação por texto e/ou por fala, é essencial a existência de sistemas capazes de traduzir dados internos em frases, ou mesmo textos, que possam ser consultados, pelos seus destinatários, no ecrã, ou eventualmente, convertidos em som para serem ouvidos. É essencial que esses sistemas de geração de linguagem natural (GNL) gerem frases nas línguas nativas dos utilizadores (no nosso caso, português), e que esse desenvolvimento seja o mais fácil possível e que o seu resultado seja percecionado como natural. A criação de um sistema GNL não é uma tarefa fácil, mesmo para um pequeno problema. As principais dificuldades estão relacionadas com: o facto das abordagens clássicas serem muito consumidoras de tempo de desenvolvimento e requerem um conhecimento muito profundo dos desenvolvedores; a falta de variabilidade na geração das frases, em muitos dos métodos de geração; a falta de ferramentas de desenvolvimento, facilmente acessíveis; a escassez de recursos; o reduzido número de línguas suportadas. O objetivo principal foi o de propor, desenvolver e testar um método para conversão da Dados-para-Português, que pudesse ser desenvolvido com uma pequena quantidade de tempo e de recursos, mas, mesmo assim, capaz de gerar frases com variabilidade e qualidade. A tese defendida é que esse objetivo pode ser alcançado adotando uma abordagem orientada à geração de linguagem, baseada em dados – mais precisamente, geração de linguagem baseada em tradução – e seguindo uma Metodologia de Investigação em Engenharia.
Nesta tese, são apresentados dois sistemas GNL, baseados em Dados-para-Texto. Eles foram especificados para providenciarem, rapidamente, o desenvolvimento de um sistema GLN que possa gerar frases com boa qualidade. Esses sistemas utilizam ferramentas disponíveis gratuitamente, e podem ser desenvolvidos por pessoas com poucos conhecimentos linguísticos. A sua principal caraterística é o uso de técnicas de tradução automática baseadas em modelos estatísticos. Essa abordagem requer apenas um pequeno corpus de linguagem natural, tornando fácil, e barato, o seu desenvolvimento.
O principal resultado desta tese é a demonstração que é possível a criação de sistemas capazes de efetuar a tradução de informação/dados em frases, em português, com qualidade decente. Isto é feito sem um grande esforço no que diz respeito à criação de recursos, e com o conhecimento comum de um experiente desenvolvedor de aplicações. Os sistemas criados, particularmente o sistema híbrido, são capazes de fornecer uma boa solução para problemas de conversão de dados para texto.

Keywords data-to-text, natural language generation, automatic translation, phrase-based translation, evaluation, Portuguese

Abstract

To enable the interaction by text and/or speech it is essential that we devise systems capable of translating internal data into sentences or texts that can be shown on screen or heard by users. In this context, it is essential that these natural language generation (NLG) systems provide sentences in the native languages of the users (in our case European Portuguese) and enable an easy development and integration process while providing an output that is perceived as natural. The creation of high quality NLG systems is not an easy task, even for a small domain. The main difficulties arise from: classic approaches being very demanding in know-how and development time; a lack of variability in generated sentences of most generation methods; a difficulty in easily accessing complete tools; shortage of resources, such as large corpora; and support being available in only a limited number of languages.
The main goal of this work was to propose, develop and test a method to convert Data-to-Portuguese, which can be developed with the smallest amount possible of time and resources, but being capable of generating utterances with variability and quality. The thesis defended argues that this goal can be achieved adopting data-driven language generation – more precisely generation based in language translation – and following an Engineering Research Methodology.
In this thesis, two Data2Text NLG systems are presented. They were designed to provide a way to quickly develop an NLG system which can generate sentences with good quality. The proposed systems use tools that are freely available and can be developed by people with low linguistic skills. One important characteristic is the use of statistical machine translation techniques and this approach requires only a small natural language corpora resulting in easier and cheaper development when compared to more common approaches.
The main result of this thesis is the demonstration that, by following the proposed approach, it is possible to create systems capable of translating information/data into good quality sentences in Portuguese. This is done without major effort regarding resources creation and with the common knowledge of an experienced application developer. The systems created, particularly the hybrid system, are capable of providing a good solution for problems in data to text conversion.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

In this chapter we start by presenting the motivations that guide our work. We describe the problem and the goals that we want to achieve. At the end, we justify the methodology used, and finish with the presentation of publications related to the work presented here.

1.1 Context and Motivation

Since immemorial times humanity has felt the need to communicate. First, using voices. Later, with drawings on rocks and cave walls. Much later, through written documents. And it seems that the need for communication is increasing. See, for instance, the ‘explosion’ in the number of people who every day use ‘social networks’. This is only possible thanks to computers, or related devices, bringing new challenges and opportunities. And, let us do not forget that computers, at first, were only used for military, research or business purposes. Ordinary people did not benefit from using them directly, at the start. It was only with the emergence of the Personal Computer that their use began to be democratized.

Interaction and Language

Interaction with computers (or other similar systems) is essentially the “transmission of *data* or, even, *information*”. Human languages, named *natural languages*, are the best way we humans developed for encoding data and reason, as stated by Santos (1992) “Natural language is so far the most comprehensive tool for (humans to) encode and reason with knowledge”.

By ‘natural language’ we mean any kind of output similar to what humans can do – *written text* or *speech*. Electronic devices express data as numbers, which are the natural form, but humans understand text better text than numerical data, even when it is expressed as graphs or any other kind of graphic output (Law et al., 2005).

Our interaction with devices is generally bidirectional. As an example, let us look at one application that provides us with information about weather conditions, available on a smartphone. First, we search for a country, a city or date, supplying data to the application. Then, the same application can give us an answer with the weather forecast. It can be supplied in natural language, in the form of a small text or even with the reading of that text, by a speech synthesizer (two samples of what is generally designated as ‘output modality’ in a context of multimodal scenario (Bernsen, 2007)).

In order to develop applications such as these, it is necessary to develop technologies capable of translating data from electronic devices into texts, speech, images, etc. However, it is not enough to produce these technologies. It is also necessary that applications have the ability of choosing the best way to send data to users.

New devices, such as *smartphones* and *tablets* have brought about a small (or perhaps, big?) revolution in the way people interact with technology, creating new challenges and opportunities. These new devices are multimodal, by default. Out of all available modalities, we are particularly interested in interaction through speech and text. For interaction through text and/or speech (as the preferred means of interaction on a two-way human–device relation) to be possible it **is essential to create modules capable of translating internal data into sentences or, preferably, into texts** to be seen on a screen or heard (if a synthesizer is used) by users.

It is also essential that these modules are able to produce sentences or texts in the native languages of users. In our case, we are interested in Portuguese. Another requirement is that the development process must be as easy as possible, demanding the smallest possible investment of time and money. Lastly, the quality of output is very important, and it must be as natural as possible to human users.

Converting data into sentences

In order to express the meaning of some data source, the automatic production of sentences, can be done through the use of Natural Language Processing (NLP) techniques. NLP is related to the processing of human languages by computers. It is a field of both artificial intelligence and computer science. NLP is almost as old as computers themselves (Bates, 1995), and the publishing of Alan Turing’s paper “Computing

Machinery and Intelligence" (Turing, 1950) marked the birth of what we call 'artificial intelligence'. The first effort was the direct translation between different languages. It was quickly discovered that processing human languages (known as 'natural languages') is a more complex task than was initially expected.

Natural Language Generation (NLG) is one of the techniques used, and it is defined (ex: (Reiter and Dale, 1997)) as the process of constructing natural language output, in English or other human languages, from a source of non linguistic input. Despite being recent, NLG systems are far from being considered basic, requiring many resources for their development. These systems are expected to decide "How to say", after deciding "What to say" (Lemon, 2011, Bateman and Zock, 2004). This means that NLG systems should aim to emulate humans, producing utterances which are syntactically and semantically correct, as well as contextually appropriate.

Early efforts in the field of NLG were the interactions made by the use of sentences, texts or pre-prepared speeches. Jurafsky and Martin (1999, Chapter 20) argue that the first NLG program was probably the C program "Hello, world" – and indeed, it does generate a well formed sentence in English, whenever the program is executed. These type of systems, called *canned text*, are easy to implement, but they need the intervention of a programmer whenever a modification is necessary. An evolution from canned text was the creation of *template filling* systems. Here, the system maps some non-linguistic input directly onto gaps previously prepared in their output (Reiter and Dale, 1997, p. 83-84). This approach can be seen, for instance, in certain letters that we receive, where everyone receives the same content and the only difference is the name or address. Another example, using speech, are the spoken announcements at railway stations or health clinics, announcing the arrival of the next train, or calling a patient to a medical appointment. Both of these systems work with *templates*. One of their major advantages is the speed and ease of development. Their main drawback, however, is the lack of variability in the output produced, which can only be corrected through a high investment in production. Without that variability, human users tend to consider the produced output as non-natural, which reduces their acceptance.

As a consequence, a new kind of systems for generating sentences and texts has been developed. They are commonly known as classic Natural Language Generation systems (Reiter and Dale, 1997, Reiter and Dale, 2000), doing their job in three steps (Reiter and Dale, 2000, p.47–72)(Reiter, 2010, p.579–586):

1. Document Planning (makes a specification of text structure, using domain knowledge about what information is suitable for the proposed communicative goal);

2. Microplanning, including Lexicalisation (choosing the best words to describe what has been decided in document planning), Aggregation (aggregates the structures created by document planning in sentences and paragraphs) and Referring Expressions Generation (task that chooses referring expressions to identify domain entities);

3. Surface Realization (task that executes all that has been decided, and produces the final sentence or final text).

These systems usually use very strict rules related to their domain, and rely heavily on developers' knowledge and experience. Despite the existence of several successful systems (Hunter et al., 2005, Konstantopoulos et al., 2008, McCauley et al., 2008), the resources needed to develop classic NLG systems remain scarce and their development takes a lot of time. Those involved must have a deep knowledge of Linguistics and Natural Language Processing. Another limitation is the difficulty in adapting the systems which are developed to new domains (Lemon, 2011).

These constraints led to the appearance of new approaches. One new idea which has been proposed is the use of systems which use Machine Learning. The goal is to train/teach a system with previously known data. Then, with that knowledge it is possible to infer new data. Usually, the inferred data does not belong to the training set.

Several approaches have been followed. Some use ontologies as a structure to collect corpora and then produce new utterances (Stent et al., 2004, Stent and Molina, 2009, Vogiatzis et al., 2008); others design their systems as a translation process, using an aligned corpus with an input language and an output language (Langner and Black, 2009, Langner, 2010). Some of them use Moses (Koehn et al., 2007) as a translation tool. Each system has its own characteristics, and their development can be quite different. Not all of them require a full NLG system to be developed. Many systems have the initial part resolved (i.e. they already know 'what to say'). This results in a much simpler system, since they only have to develop the 'how to say' part. In most cases, the data they need to transmit originates from a database query. In these cases, we are in the presence of a Data-to-Text system (Reiter, 2007). This kind of NLG system has some differences from other kinds of natural language generation systems (Langner, 2010). In this case, we practically only use the last step of the generation process – Surface Realization.

1.2 The Problem(s)

The creation of NLG systems is not an easy task, particularly for developers without a deep knowledge in fields such as Linguistics. On the other hand, experts in Linguistics generally do not have knowledge of programming languages, to be able to program theses systems. This leads to a problem which is difficult to solve. The use of multidisciplinary teams can be a solution, but that is not necessarily easy or a certainty. Bringing together people with multiple skills involves considerable costs. The definition of system requirements, training of the people involved, or the development process itself, mean the amount of time needed to develop such systems is also an issue. One possible way to make the process cost-effective, is to define and create modules which can be reused in several applications.

Even with that knowledge, developers can come across several difficulties. Every language has its own idiosyncrasies. Most of them have very complex rules, at the level of syntax or spelling, which makes the task of programming them very difficult. Portuguese is known for being one of these languages.

The number of Portuguese speakers is increasing. It is now the fourth most spoken language in the world, with 261 millions speakers, and is expected to rise to more than 387 millions speakers by 2050 (Agência Lusa, 2010, Amado et al., 2016). These numbers justify investing in this kind of systems for Portuguese, and makes them potentially profitable.

When analyzing literature in this field, we become aware that the majority of systems and resources are directed at the English language. Since the differences between English and Portuguese are substantial, it is clearly necessary to explore new ways of producing NLG systems for the Portuguese language, using as few resources as possible.

A feature that all of these systems must have is the ability of produce sentences and texts, with enough variability, over time. Let us keep in mind that variability is a major characteristic of human conversation. If that is not taken into account, the sentences produced by systems may be considered 'boring'. That may lead to a situation where their use is not perceived as interesting or useful.

In short, the main difficulties encountered result from:

- The lack of tools. Most available tools resulted from PhD projects. Their use is usually restricted to the original purpose behind their development. Adapting them is difficult, if not impossible, in most cases. The lack of documentation is another major obstacle faced when attempting to use them.

- The lack of resources. This item is directly connected to the previous one. It is quite difficult to find good corpora, grammars, or other resources. For instance, during the entire time this work was being carried out, we had great difficulty finding corpora in Portuguese. And when we did find them, they were hard to access, or in poor quality. The lack of variability was, and still is, also a concern.

- Classic approaches, such as the ones proposed, for instance by Reiter (Reiter and Dale, 2000), are quite demanding in terms of know-how and time. As mentioned at the beginning of this section, one of the major difficulties of this approach is the high cost of developing such a system. This cost has to do with the involvement of highly skilled people and the resources required.

- A lack of variability in generated sentences. People like to speak and interact with each other and they are highly creative in doing so. That is quite obvious in the oral, or written, sentences they produce. That creativity is what every NLG system should aim to achieve. The use of 'templates' is a possible solution to produce a NLG system. Their use, however, results in a lack of variability in the sentences which are generated.

- Lack of coverage of many languages. We found many resources and tools that use English as the language of destination for NLG systems. Other languages are not so lucky – resources in Portuguese are scarce and difficult to use.

1.3 Thesis Statement

In order to increase the availability and usefulness of Data-to-Text systems, it is necessary to develop methods which make it possible to address new domains with little or no resources and with low knowledge requirements for developers. Our Thesis is that this can be done by using data-driven systems.

A major reason for defending this approach is the potential for creating data-to-text systems for new domains, with the very scarce resources available, by developers with reduced knowledge in language syntax and semantics. Another very important aspect to be considered is the development time required, since NLG systems, due to their complexity, tend to take a long time to be developed. Data driven systems will also provide some variability and naturalness to the sentences produced, which is essential in order to engage the interest of potential users.

In this thesis, we present two Data-to-Text NLG systems. They were designed, as described above, to provide a way of quickly develop an NLG system which can generate high quality sentences. These systems use freely available tools and can be created by developers without a University degree in Linguistics or Computational Linguistics. One major characteristic is the use of statistical machine translation techniques. This approach requires only a small natural language corpora, making development easier and less costly. The process which was developed uses several modules. The first one is responsible for the production of utterances. The second is a module capable of scoring the quality of those utterances. The hybrid approach of our system is complemented with a templates module that replaces a rejected sentence, by a new, better one. Our second system addresses more demanding scenarios. It was used to prove the value and accuracy of our proposal. It is worth noting that the utterances provided by this system can be used in the form of written text or with a speech synthesizer, allowing it to be applied to multimodal systems.

1.4 Objectives

Keeping in mind the thesis statement above, our main goal is to propose, develop and test a method of converting Data-to-Portuguese, which can be developed with the smallest possible amount of time and resources. Our proposal intends to generate utterances with variability and quality.

Some specific goals we aim to achieve:

- To specify and develop systems with our approach for different domains – we intend to develop, at least, two different systems to test the effectiveness of our proposed approach;

- To evaluate the utterances produced – sentences produced by each system are only usable if they are correct. This evaluation should be done through the use of automatic metrics (so it is possible to make a comparison with other systems) and by human evaluators (effectively the final target of the produced sentences);

- To generate new sentences, if evaluation results in a lower score – it is expected that a small part of the sentences produced will be rejected. If this occurs, they should be replaced by new ones, expressing the same data and meaning. They should also be grammatically correct;

- To integrate our systems with ongoing projects, if possible – sometimes, good proposals fail when tested with real problems. It is our intention to apply the proposed systems to real problems so as to assess their usefulness.

1.5 Method

The American National Academy of Engineering (1995, p. 3–4) express the difference between 'basic scientific research' and 'engineering research'. The first is expected to be concerned with the "discovery of new phenomena and their integration into coherent conceptual models of major physical or biological systems". This approach is expected to find new developments and findings to problems, even if the usefulness of proposed solutions is not yet evident. For instance, see the research developed by Mandelbrot (Mandelbrot, 1989) about fractals, which was only put to use many years later. Whereas the second is expected to find new solutions to problems, but the solution must be aimed at solving a current problem of society.

Nallaperumal and Krishnan (2015, p. 49-50) argue that there is no contradiction between these two concepts of research. They complete each other, and often a scientist acts as an engineer and vice-versa. Most of the time, the problems that Engineering Research deals with are complex, and do not have a simple solution. The methodology used intends to propose an initial solution, and then iterate it, to find a better one.

The work carried out in this thesis follows an Engineering Research Methodology. The choice of this methodology is motivated by the nature of the problem we intend to solve – the development of an NLG system, using Portuguese. This is not a 'scientific' problem, but an 'engineering' problem, since there are several solutions available, but none, in our opinion, are good enough to process the Portuguese language. First, the problem is going to be described. Second, possible approaches are described. Finally, we will try to prove our approach to solving the problem.

1.6 Publications

The work presented in this thesis resulted in several publications:

1. The first version of the generation system, making use of Moses and for the domain of medication intake assistance, was published in:

    - Pereira, José Casimiro and Teixeira, António. 2015. Geração de linguagem natural para conversão de dados em texto - aplicação a um assistente de medicação para o português. In *Linguamática*, 7(1):3–21, Julho, 2015. [1]

- Teixeira, António; Pereira, José Casimiro; Francisco, Pedro and Almeida, Nuno. 2015. Tradução automática na interação com máquinas. In *Oslo Studies in Language*, 7(1) – Linguística, Informática e Tradução: Mundos que se Cruzam, pp.283–300. [2]

2. Evaluation Module and the proposal of an **Hybrid approach** combining the first system with a template based generation, described in:

    - Pereira, José Casimiro; Teixeira, António and Sousa Pinto, Joaquim. 2015. Towards a Hybrid NLG System for Data2Text in Portuguese In *Proceedings of 8th Iberian Conference on Information Systems and Technologies (CISTI 2015)* pages 679–684, Águeda, Portugal.

3. The second version of the generation system, also making use of Moses and aiming at the production of short text expressing the opinions that customers give about hotels' services, was published in:

    - Pereira, José Casimiro; Teixeira, António; Rodrigues, Mário; Miguel, Pedro and Sousa Pinto, Joaquim. 2017. Natural Transmission of Information Extraction Results to End-Users – A Proof-of-Concept using Data-to-Text In *Proceedings of 6th Symposium on Languages, Applications and Technologie (SLATE'17)* pages 14, Vila do Conde, Portugal.

4. Integration with Multimodal applications, published in:

    - Pereira, José Casimiro; Teixeira, António J.S. and Sousa Pinto, Joaquim. 2013. Natural Language Generation in the context of Multimodal Interaction in Portuguese. In *Proceedings of 8th Iberian Conference on Information Systems and Technologies (CISTI 2013)* pp. 333–337, Lisbon, Portugal.
    - Ferreira, Flávio; Almeida, Nuno; Rosa, Ana Filipa; Oliveira, André; Teixeira, António and Pereira, José Casimiro. 2013. Multimodal and Adaptable Medication Assistant for the Elderly – A prototype for Interaction and Usability in Smartphones. In *Proceedings of 8th Iberian Conference on Information Systems and Technologies (CISTI 2013)*, pp. 309–314, Lisbon, Portugal, June, 2013.
    - Ferreira, Flávio, Almeida, Nuno; Rosa, Ana Filipa; Oliveira, André; Pereira; José Casimiro; Silva, Samuel and Teixeira; António. 2014. Elderly Centered

[1] `http://www.linguamatica.com/index.php/linguamatica/article/view/V7N1-1`

[2] `https://www.journals.uio.no/index.php/osla/article/view/1451/1348`

Design for Interaction – The Case of the S4S Medication Assistant. In *Procedia Computer Science*, 27 (2014), pp.398–408, Vigo, Spain, November, DSAI 2013.

- Teixeira, António; Ferreira, Flávio; Almeida, Nuno; Rosa, Ana Filipa; Pereira, José Casimiro; Silva, Samuel; Queirós, Alexandra and Oliveira, André. 2013. Multimodality and Adaptation for an Enhanced Mobile Medication Assistant for the Elderly. In *MOBACC 2013 - 3rd Workshop on Mobile Accessibility @ CHI2013*, Paris, France, April, 2013.

5. Fast adaptation of translation based NLG/Data-to-Text modules to other domains, using Information Extraction –with application to tourism domain–, publish in:

    - Teixeira, António; Miguel, Pedro; Rodrigues, Mário; Pereira, José Casimiro; and Amorim, Marlene 2016. From Web to Persons – Providing Useful Information on Hotels Combining Information Extraction and Natural Language Generation. In *Proceedings of the IberSPEECH 2016 Conference*, Lisbon, Portugal. November 2016.

6. Other, more generic, publications on NLG and multimodal interaction:

    - Teixeira, António; Ferreira, Flávio; Almeida, Nuno; Silva, Samuel; Rosa, Ana Filipa; Pereira, José Casimiro, and Vieira, Diogo. 2016. Design and development of medication assistant: older adults centred design to go beyond simple medication reminders. In *Universal Access in the Information Society*, pages 1–16. doi:10.1007/s10209-016-0487-7

    - Teixeira, António; Almeida, Nuno; Pereira, Carlos; Oliveira e Silva, Miguel and Pereira, José Casimiro. 2013. Serviços de Suporte à Interação Multimodal. In António J. S. Teixeira, Alexandra Queirós, e Nelson Pacheco da Rocha, editors, *Laboratório Vivo de Usabilidade – Living Usability Lab.* ARC Publishing e Editores, Chapter 11, pp. 151–165.

    - Teixeira, António; Pereira, Carlos; Oliveira e Silva, Miguel; Pacheco, Osvaldo; and Pereira, José Casimiro 2011. AdaptO - Adaptive Multimodal Output. In *1st. International Conference on Pervasise and Embedded Computing and Communication Systems (PECCS 2011)*, pages 91–100, Vilamoura, Algarve, Portugal.

1.7 Thesis structure

The remainder of this thesis is structured as follows:

In the next chapter, the second, we present state-of-the-art work related with our thesis. We discuss the importance of multimodal systems, with special focus in output modality. We also discuss what NLG systems are, and their variations.

Our first system is described in Chapter 3. The definition of input and output languages, the definition of sub-systems and their evaluation are explained here.

Going a step further, the hybrid approach is described in Chapter 4. Besides the presentation of the templates module, we present the module which evaluates produced utterances.

In Chapter 5 is presented our approach with a second system. This time, the domain was the evaluation to services provided by hotels. We describe the corpora needed and evaluate the influence that out-of-domain corpus has on the language model.

Finally, in Chapter 6, we discuss the work carried out and the main results, and present several suggestions for future work.

Chapter 2

Background and Related Work

In this chapter, we present a brief overview of technologies related to our work. First, we start by defining what is multimodal interaction. We follow up with the description of Natural Language Generation, its types of approaches and the description of a statistical translation tool used to build NLG systems. The evaluation of utterances produced, based on automatic metrics and human evaluators, is also described. We finish by presenting state-of-the-art work related to these areas, with the description of relevant systems based on Data-to-Text and based on Templates.

2.1 Background

Interaction with users is a major concern of this work. We are particularly interested in output, since research of the literature seems to show that research on input modalities has a special prominence and there is a lack of studies on output modalities. Another motivation is the fact that we are working on several projects where multimodal output is the best choice for user interaction (in this case, usually elderly users) and where the input is simple (Teixeira et al., 2011b, Teixeira et al., 2011a, Ferreira et al., 2014, Teixeira et al., 2013a, Teixeira et al., 2013b). As a result, in this section, we start by describing multimodal interaction and the need to work on output. Later, we present a brief overview of Natural Language Generation (NLG) and its subtypes. The automatic metrics most often used to evaluate NLG systems are also presented.

2.1.1 Interaction / Multimodal

The issue of how people interact with computers, or other electronic devices, is not new. For instance, Reeves and Nass (1996, Chapter 2) wonders, if computers are electronic machines, why do people tend to be polite when they interact with them? On other hand, Nass, Fogg and Moon (1996) states that people can see computers as their 'teammates'. However, humans and computers do not speak the same 'language'. This has led to a problem that to this day has not been solved. Humans expect computers, or other kind of similar devices, to give them what they need, and on the other hand, computers need to know what humans want them to do. Sebe (2009) argues that in traditional Human-Computer Interaction (HCI), the interaction is between one human being and a computer, in a single-task environment. Meanwhile, the appearance of new tools, facilitating, for instance, collaborative work, are transforming that reality. Most of them involve the collaboration of several users and interaction does not always resort to the use of written commands. This lead to the following question - if humans, when relating to each other, interact in multiple ways (voice, text, touch, etc.), why can they not do the same with computers?

The answer to that question is complex, and does not have a single answer. One possible hypothesis is to give users the ability to interact with devices in multiple ways, developing a *multimodal system*. Bernsen (2007) and Sebe (2009) define a multimodal system as one that receives input from users, simultaneously, in multiple ways, or gives feedback to users in multiple ways. Interaction takes place through a physical medium, which requires the use of human senses. The senses of sight, hearing or touch are the senses most commonly used. Nevertheless, smell and taste are starting to look like a plausible way of communicating too (Legin et al., 2005). Multimodality need not be intrusive and it is expected it will not be. Users do not need to realize multimodality, nor the use of multimodal media (Bernsen, 2007).

The use of multimodal interfaces tends to represent human ways of interacting. If, on the one hand, this can be positive, as humans tend to appreciate it better (Oviatt, 1997, Oviatt, 2003), on the other hand, the design and development of such systems can be harder than those designed to be unimodal. Dumas, Lalanne and Oviatt (2009) argue that the increases in productivity and efficiency, when using multimodal interfaces, are not expressive, in comparison to unimodal interaction. Nevertheless, they do produce a reduction in errors, as well as improving reliability and the handling of errors, by using different means of communication to validate what is sent to or received from users (Oviatt, 1997, Dumas et al., 2009).

The design of multimodal interfaces has characteristics that are specific and not present in traditional HCI (Sebe, 2009). Focusing on the machine side, multimodal architecture, which is presented in Figure 2.1, is composed of four modules: fusion engine, fission engine, dialogue manager and context manager. As the name suggests, the fusion engine collects the inputs provided by the several means used. Its job is to make a common interpretation of facts provided by input devices. Interpretation made by the fusion engine are communicated to the dialogue manager. The dialogue manager is the 'heart' of multimodal interaction. It has the responsibility to identify the dialogue state and identify the transition needed. After receiving the input, it must be communicated to the application or applications which are using the multimodal interface, to be processed. When it receives the communication to be presented to the user, it delivers the message to the fission engine. The fission engine determines the best output device to present the output message to the user. If necessary, it can use more than one simultaneously. The context manager is used to identify the user profile or the context where the communication is going to take place. It works closely with the other three modules, and can influence their behavior (Dumas et al., 2009).

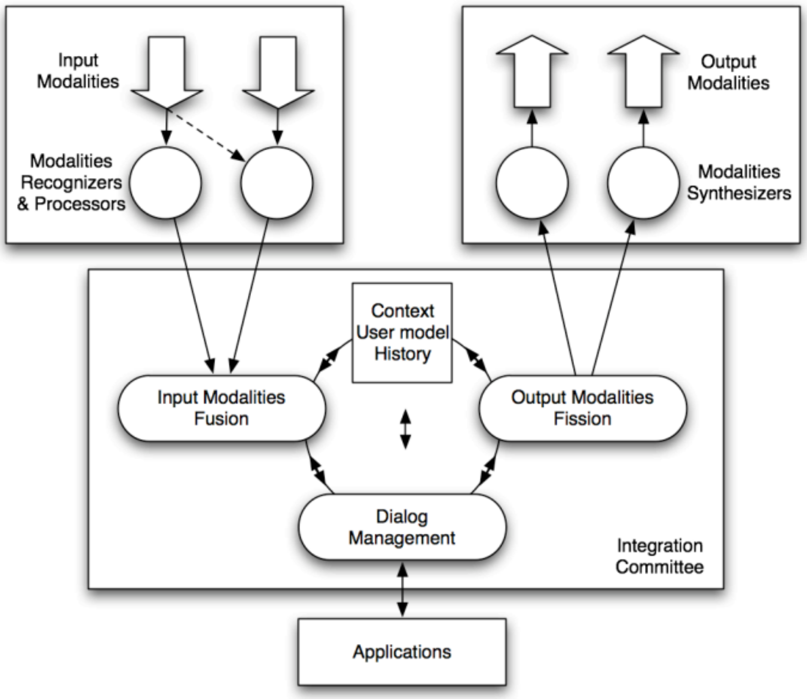


Figure 2.1: Multimodal architecture, retrieved from (Dumas et al., 2009)

This architecture, as research of the literature suggests, is well accepted by the research community. However, it seems to treat the input and output devices as dummies, where their only responsibility is to receive the input data, and send it to the system, or receive output messages already adapted to the context and user. In this approach the fusion and fission engines are required to be very knowledgeable of all the available input and output devices, making it potentially very complex, and more difficult to scale and extend applications to new input or output devices. This problem is particularly important in output devices since they convey the information to users, closing the interaction loop with the application. An alternative approach, presented by (Teixeira et al., 2011a), aims to enhance the intelligence of output devices, making them able to adapt to a dynamic context and to users. In this way, the responsibility to ensure adaptability in the applications is not centralized in a potentially very complex and knowledgeable fission engine, but is shared with the output devices themselves.

Multimodal interfaces have attracted the interest of many researchers. The first reported multimodal system was, probably, the “Put-That-There” system (Bolt, 1980), where a user in a ‘media room’, could interact with an image projected on the wall, through their voice and/or gestures, changing the position of elements represented in that image. From 1980 till today many systems have been introduced. As an example, we named two recent multimodal systems, that are applied to a very different scenarios. First, the use of holographic 3D images to command robots (Noor and Aras, 2015). Second, the use of multimodal interfaces in the control room of a modern factory (Heimonen et al., 2016).

2.1.2 Output matters! The need to transmit system information to users

The multimodal output system should not be a system with a collection of output devices or output modalities. Its use must correspond to users’ needs. The system must identify what information needs to be presented, which interaction components are available, the context where the information is going to be presented and how the available modalities should be used (Rousseau et al., 2006). For instance, suppose that a user has an application on their smartphone, which must remind them to take a certain medication. Suppose that they are at a classical music concert, and an alert message needs to be produced. The application should realize that the sound channel should not be used, unless it is a very, very urgent message. On other hand, the same application, if the user is at a football stadium, should realize that the sound channel needs to be used loudly, in addition to a written text.

When two or more people are talking, each one of them can identify if their message is not understood by the other participants. This awareness causes them to change the way of speaking, or even to change the message. Karpov and coworkers (2008) and Mairesse and Walker (2010) argue that if a dialogue manager adapts quickly to users, in the same way that humans do, it will be more effective at its job. According to Coetzee, Viviers and Barnard (2009) , most applications were developed with the goal of satisfying the average user. This means that these applications do not really have the ability to adapt to users. Therefore, it is hard to consider them multimodal applications.

One possible way to provide good adaptability to users can be the use of natural language, in order to create a more natural and intuitive interface. Nevertheless, before one can decide on the best interface, we must realize that each potential user has different abilities and handicaps that must be understood, in order to design a good interface. Coetzee, Viviers and Barnard (2009) argue that is better to concentrate on the user's abilities rather than on their handicaps. If the application is aimed at abilities, it can provide the best possible modality for user interaction. Figure 2.2 presents a list of abilities and assumptions that applications are able to make about users. For instance, it is possible to assume that if a user can read, they must see.

Ability associated with output	**Assumptions**
Can See	None
Can Hear	None
Can Read	User can see
Can Read (simplified text)	User can see
Can Understand South African Sign Language	User can see
Can Feel	None
Can Understand Braille	User can feel
Can Lip Read	User can see

Figure 2.2: List of human abilities linked with system assumptions about the best possible output, retrieved from (Coetzee et al., 2009)

In addition to what a user can do, it is very important to also understand what a user would like to do, i.e. their preferences. A user's preferences express how they best understand and absorb information, using the five human senses. Figure 2.3 presents a list of human characteristics, connected to those senses, that most influence interaction with computers. Sight, hearing and touch are referred to here. The term *visual* is associated with sight; *aural* is associated with hearing; *read/write* is associated with sight and hearing; and *kinaesthetic* is associated with touch sense and refers to all body movement. If users have different preferences, this will certainly have an impact on their expectations of computer interfaces.

One major aspect of these preferences, particularly when the *visual*, *aural* or *read-/write* interaction is chosen to provide information to users, is that the `data-to-text`

Perceptual Preference	**Description**
Visual (V)	Individual prefers pictures, graphs and diagrams
Aural (A)	Individual prefers spoken words
Read/Write (R)	Individual prefers reading and writing texts
Kinaesthetic (K)	Individual prefers to move his/her body and manipulate things with his/her own hands

Figure 2.3: Perceptual preferences of users, retrieved from (Coetzee et al., 2009)

approach can become prominent. Since the use of databases is increasing, data-to-text systems can provide texts (or simple messages) based on data stored in databases, which can be sent to users as oral or text speech.

One last item to mention is the literacy level of application users. This aspect is harder to assess than the two mentioned above. Coetzee, Viviers and Barnard (2009) identify five categories to describe it. First, *illiterate* identifies a person who does not know how to read or write; *cultural* refers to a person who do not understand the icons, expressions and role models associated to a language; *grammatical* identifies a person who uses grammar incorrectly; *second language* refers to a person that is fluent in their mother language, but not fluent in the interface's language; and *deaf* identifies a person that is fluent in Sign Language, but has great difficulty interacting with oral or sound messages. Alongside these categories, it is necessary to factor in people's cultural traits, since they can also shape the way that they perceive the application interface.

2.1.3 Natural Language Generation: A brief overview

Natural Language Generation (NLG) is a subfield of Artificial Intelligence and Computational Linguistics. Its major focus is the production of comprehensible texts in human languages. Lemon (2011) , in short, defines NLG systems as a system that must define *how to say* something, after someone defines *what to say.* Most of the time, systems receive, as input, a non linguistic source of data and, using their knowledge of language and application domain, transform it into a relevant report, document, help message, instructions or other text output (Reiter and Dale, 2000, p.1). The use of the word 'text' must be understood in a broad sense. The output can likewise be sent to users as speech. The only requirement is that a proper speech synthesizer must exist (Reiter and Dale, 1997). In this case, when speech is used, prosody is also a concern.

Although the earliest studies on NLG date back to the 1970s, with the appearance of what is considered to be the first PhD thesis on NLG (Goldman, 1975), the development and study of this subject really only began later, particularly in the 1990s (Reiter and Dale, 2000, p.19-20)(Ramos-Soto et al., 2016).

As expressed before, the aim of NLG systems is the production of output which is useful to its users. This leads to some relevant questions, related to computer–human interaction and usability. How should NLG systems interact with humans? How should they transmit data to humans? What behavior should humans expect computers to have? What is considered 'readable' or 'understandable'? How can computers convert large amounts of data (sometimes numerical data) into representations users can understand? (Reiter and Dale, 2000, p.2)

The previous considerations must be taken into account when one decides to develop an NLG system. NLG systems are not always are the best solution to fulfill the needs of users. Sometimes it is better to present data in a graphical format, or other kinds of output, rather than in a text format (Reiter and Dale, 2000, p.24). The definition of NLG systems is so vague, that almost any system that produces some sort of textual output can be considered an NLG system. Canned text systems, on one hand, are the simplest of them, and on the other hand, full NLG systems, as described by Reiter and Dale (2000, p. 60), are the most complex. Nevertheless, not every user will understand the usefulness or need for a system as sophisticated as NLG systems are. Most of the time it is cheaper and faster to develop simpler systems to produce textual outputs (Reiter and Dale, 1997)(Reiter and Dale, 2000, p.25).

Types of Approaches

As argued before, there is a large variety of types of systems with different purposes. One can categorize NLG systems into three main types of approach: template-based NLG, conventional NLG and, more recently, trainable generation NLG (Lemon, 2011). These classes should not be seen as 'closed boxes', since systems can be classified in more than one category.

2.1.3.1 Template-based NLG

Templates are one of the oldest, if not the oldest, ways of producing Natural Language through automatic systems. The first such system on record is probably the *IFPS* (Glahn, 1970a, Glahn, 1970b). This system was one of many designed for weather forecasting.

Template based systems are defined as natural language generating systems, that map their non-linguistic input directly onto the linguistic surface structure (Reiter and Dale, 1997, p.83–84). Usually, this linguistic structure has gaps that must be properly filled with data, to generate a usable output utterance. Using the example provided in (Deemter et al., 2005, p.16), let us say that one wants to provide information about train departures. The data provided can be represented as:

`Departure(train`$_{306}$`, location`$_{\texttt{abdn}}$`, time`$_{1000}$`),`

stating that train 306 leaves Aberdeen at 10:00 am. This input data can be obtained from a database or another source. Then, if the template is used

`[train]` *`is leaving`* `[town]` *`now`*

the gaps `train` and `town` must be filled with proper data from the input data, in order to generate a useful utterance.

Lemon (2011) argues that this type of system is mainly used in industrial dialogue systems, where the main concern is the rapid development of affordable, quality interfaces which provide the necessary output. The major disadvantage of templates is the need to develop a new template, whenever the covered features change.

Templates do not seem to have been a consensual NLG technology. Several authors have questioned and/or defended them (Reiter, 1995, Busemann and Horacek, 1998, Deemter et al., 1999, Deemter et al., 2005). From our perspective, this is a useless argument. *Templates*, *conventional NLG* and *Trainable NLG* systems each have their own advantages and flaws, which must be understood in order to use them effectively.

2.1.3.2 Conventional NLG

Since the early development of NLG systems, researchers have proposed several architectures for NLG systems, differently from 'templates'. De Smedt, Horacek and Zock (1993) identified several architectures, related to the development of NLG systems. The existence of different architectures makes development harder, since it makes the exchange of tools or resources between systems more difficult. Mellish et al. (2006) argue that the appearance of Reiter's work (Reiter, 1994) marks the beginning of the attempt to settle on a common architecture for NLG systems. In that work, Reiter summarizes the most prominent systems at the time, and concludes that almost every system has the same base structure. Reiter and Dale (2000, p.59,60), (reinforced by Reiter (2010)) presented a proposal for the architecture of NLG systems. In their proposal, systems are divided into three main modules: Document Planning, Microplanning and Surface Realization. In the following subsections, a brief description of each of these modules is given. Figure 2.4 shows Reiter and Dale's proposal.

However, their proposal was not unanimously accepted. Mellish et al. (2006) presented a survey to several systems which seemed to confirm Reiter's and Dale's architecture, at least partly. They argue that the intermediate data structures – Document Plan and Text Specification – do not accommodate all tasks executed by NLG systems, and, in some cases, some tasks are incorrectly assigned to the proposed modules. Nevertheless, in the absence of a better proposal, Reiter and Dale's architecture is commonly accepted, as research of papers published on the subject of NLG systems confirms.

Document Planning

Document Planning is the first phase of NLG Systems (Reiter and Dale, 2000, p.59)(Reiter, 2010, p.579). After the goal of communication has been determined, this module has the responsibility of deciding 'what information needs to be communicated' (content determination), and 'how to organize it' (document structuring). The work of this module typically involves analyzing and defining what data is important and how to use and organize it. Its output is structured as a tree. Internal nodes specify the structure of desired output utterances at the end of the process. The leaves specify the particular data that it intends to transmit. This module is probably the most important and difficult module of Reiter and Dale's architecture, as the domain of different NLG systems can vary substantially, and it is hard to adapt a structure to all possible systems.

Microplanning

This architecture is organized as a *pipeline*. Consequently, the input of the Microplanning module is the output of the previous module. It takes the *document plan* and fills in its gaps, because the document plan is an open document, with many possible combinations of the final utterance. To complete its job, Microplanning should perform three tasks:

- **Lexicalization** – chooses what words should be used to correctly express the domain's concepts and the data that is meant to be conveyed. This is a very important task, especially if one wants high-quality output texts;

- **Aggregation** – chooses if output should be a sentence with a simple message, or if it should aggregate several messages in the same sentence. This is a very difficult task, since the domain, NLG system, and user perception or ability to read the final text can condition its production;

- **Syntactic choice** – chooses syntactic structures, such as verb tenses, temporal expressions or prepositions of place;
- **Referring expression realization** – chooses how to refer to the entities involved in the final utterance, i.e. using the entity's name or possessive pronouns, instead.

Surface Realization

This is the final step. Surface Realization is responsible for producing the final utterance, to be presented to the user. It uses the tree of data from *document plan* and *micro-realization* to generate it. Depending on the data provided, it can generate a single sentence or an entire paragraph. This is, probably, the most implemented module in NLG systems, since all of them need to generate some kind of output (Reiter, 2010, p.585). It has been observed that the other modules are not so frequently implemented. Some realizers support what is called *overgeneration and selection.* With this technique, the surface realizer generates several sentences, and then chooses the best one. One of most common selection mechanisms is the use of the *n-gram* language model, which is obtained from corpora related to the domain.

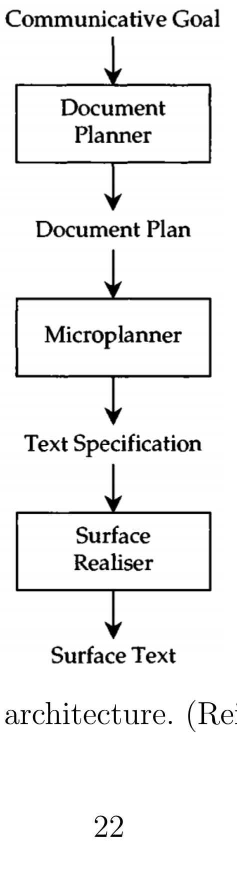


Figure 2.4: An NLG system architecture. (Reiter and Dale, 2000, p. 60).

2.1.3.3 Trainable NLG

Trainable NLG relates to the use of statistics in order to train NLG modules. Mairesse et al. (2010) argue that the development of this subfield of NLG systems has evolved into two distinct paths. The first path, uses statistics to train a model to rank candidate outputs, from a handcrafted generator. Systems using this technique use a variety of approaches. Some use n-gram language models, while others use hierarchical syntactic models. Others still, use models with a training based on utterance quality, or models trained on corpora. On the second path, systems introduce statistics at decision generation level. They train models to maximize a set of parameters with the goal of maximizing an objective function, to generate the most likely context-free derivations from a given corpus, or maximize the expected reward using reinforcement learning. These systems need a handcrafted generator to define the scope of the system, where statistics can generate an optimal solution.

In the next section we present a tool that we used in our work, to carry out the statistical training of NLG systems.

2.1.4 The basics of automatic translation

Moses[1] (Koehn et al., 2007, Moses, 2011) is a widely used implementation of the statistical approach to machine translation. This kind of systems is trained with large quantities of parallel data, and with an even larger quantity of monolingual data. The parallel data is a collection of perfectly aligned sentences, in two different languages. Each sentence from one language has a corresponding translation in the other language. The parallel data is used to teach the system how to translate small trunks of data. The monolingual data is used to teach the system what the target language should look like.

Pharaoh's successor (Koehn, 2004), Moses, uses concurrences of words and segments (known as *phrases*), from the parallel data, to infer translation correspondences between the two languages. This process has three different kind of translation models: *phrase-based MT*, *syntactic MT* and *factored MT*. Our work uses the first two, which are explained a little further ahead.

The two main components in Moses are the training pipeline and the decoder. The training is processed with several tools, some of them external to Moses. Typically, data needs to be prepared and 'cleaned' before it can be used. First, data is tokenized

[1]http://www.statmt.org/moses/

and converted to small case. Heuristic tools are then used to remove sentences that appear to be misaligned or too long. In the next step, parallel data is 'word-aligned'. These word alignments are used to extract phrase-phrase translations, or hierarchical rules, as required. A very important step is the creation of the *language model.* The language model is a statistical model created with the monolingual data, related to the target language. It is used by the decoder to try to ensure the accuracy and fluency of the output. The final step is *tuning*, where the different statistical models are weighted against each other to produce the best possible translations.

The decoder is the tool used to translate the given source sentence into the target language. This is done by searching for the highest scored sentence in the target language, using the translation model. The decoder can also output a ranked list of translation candidates. This allows the user to choose the translation that is considered to be the best.

Phrase-Based and Tree-Based translation

Moses can be trained using three different types of models: phrase-based, syntax-based or factored models (Moses, 2011). Moses uses simple correspondence between sequences of words in *phrase-based* translation. In *hierarchical phrase-based* or *syntax-based*, known as *tree-based* models, the translation uses more elaborate structures than simple correspondence. In *factored models*, the training is done with corpora tagged with part-of-speech tags.

In *phrase-based* translation the correspondence between sequences of words, known as translation tables, are the main source for the *decoder.* It uses those tables to figure out how to translate a sentence from the input language to the output language. These tables do not contain only references to single words. Usually, they contain references to small chunks of sentences, corresponding to sequences of words. That is why this translation method is called *phrase-based.* However, in this case, 'phrase' only means an arbitrary sequence of words. Each sequence of words is measured by the *n-gram* term, where `n` expresses the number of words presented in that sequence. Our work uses mainly a uni-gram sequence (one single word) and bi-gram and 3-gram sequences (sequences of two and three words, respectively).

A possible example, extracted from our scenario (presented in the next chapter) is:

```
Medicine_Type02 Medicine01 ||| MEDINX pills ||| 0.8 ||| |||
```

meaning that the bi-gram `Medicine_Type02 Medicine01`, when it occurs, has an 80% probability of being translated to `MEDINX pills` by Moses.

The translation process consists of two phases. First, the input vector is split into small blocks of words. The condition is that there is a possible translation for each block. Secondly, after translating each block, the output sentence is made, with a possible redistribution of translated blocks from their previous position. Figure 2.5 illustrates the translation process, using an input sentence from our first scenario.

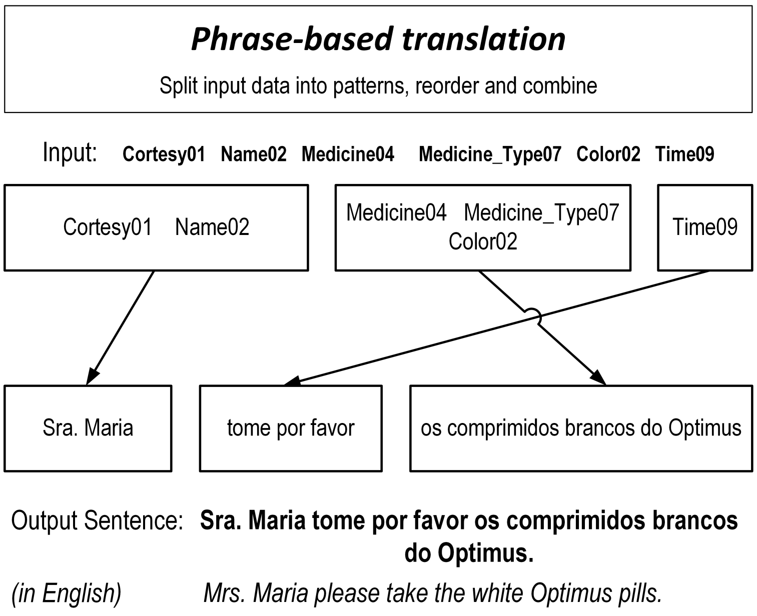

Phrase-based translation
Split input data into patterns, reorder and combine
Input: Cortesy01 Name02 Medicine04 Medicine_Type07 Color02 Time09
Cortesy01 Name02
Medicine04 Medicine_Type07 Color02
Time09
Sra. Maria
tome por favor
os comprimidos brancos do Optimus
Output Sentence: Sra. Maria tome por favor os comprimidos brancos do Optimus.
(in English) Mrs. Maria please take the white Optimus pills.

Figure 2.5: *Phrase-based* translation process

Tree-based models use a grammar consisting of SCFG (Synchronous Context-Free Grammar) rules. Whereas phrase-based models map each word directly from the input language to the output language, here, with the tree-based model, it uses and operates variables on grammar rules. When the 'output language' is processed, each term is classified as *non-terminal*, indicating that the job is not finished yet. Each non-terminal term is associated to a generic tag (eg, 'X1', 'X2', etc.) or with a tag that illustrates its content (Name, Determinant, etc.). For instance, on these syntactic models, the non-terminal terms can be tagged with linguistic labels, as verb ('VERB') or name ('NOUN'):

```
NOUN --> Medicine_Type01 ||| pills

VERB --> Time02 ||| now
```

We obtain these tags by using a *linguistic parser*. They are used then on the alignment process to produce the final sentence, with words (the *terminals*) in the output language. This model is called 'tree-model' because the input sentence is disposed in a graph tree, which is translated to a corresponding graph tree with output 'terminals'.

The tree-based model has three variants: *string-to-tree*, *tree-to-string* and *tree-to-tree*. As their names suggest, in the first variant, the input sentences remain unchanged and only the output sentences are converted to a tree; in the second the opposite happens; and in the third, all sentences, from both input and output language are converted to trees and then related. In our work we are going to use the string-to-tree approach.

2.1.5 Evaluation of the Generation results

To make the use of an NLG system possible, it is crucial to evaluate its output and determine its accuracy. However, choosing and using NLG evaluations is not yet a consensual topic among researchers (Hastie and Belz, 2014, Reiter and Belz, 2009). One major constraint is related to what users expect from an NLG system. Different users require different strategies of evaluation (Reiter and Belz, 2009). Over the last few years, several proposals have been put forward. Basically, they can be divided into two major groups: evaluation carried out by humans; and automatic evaluation, done by computers. Studies suggest that evaluations carried out by humans are usually better than those done automatically, particularly when what is evaluated is text related to support to task-based evaluation (Law et al., 2005, Reiter and Belz, 2009). Nevertheless, there are situations where that cannot be true (Law et al., 2005). Despite that fact, automatic evaluations are on the increase. That probably has to do with the high cost – in money and time – involved in human evaluation. Another relevant trait is their easy repetition (Reiter and Belz, 2009).

Reiter and Belz (2009) classified the evaluation of NLG systems in three categories: Task-based evaluation; Human-based evaluation; and Automatic metrics evaluation:

1. Task-based evaluation tries to assess the impact that generated texts have on end users. The goal is to evaluate how that text helps people in performing their tasks. This kind of evaluation takes a lot of time to organize and is very expensive (Portet et al., 2009, Reiter and Belz, 2009). Besides those constraints, task-based evaluation relies on the availability and goodwill of expert domain evaluators, which can be difficult to ensure, as Reiter and Belz (2009, p. 532) states, “such goodwill in itself is a scarce resource which must be used with care”.

2. Human evaluation is carried out by giving the generated texts to one, or preferably more people, and asking for comments on the accuracy and usefulness of those texts. This method is usually cheaper and quicker than the previous

method, since it does not require the cooperation of domain specialists. Another reason that has led developers to use this evaluation method over the task-based method, is that the latter is deeply connected with a specific task and the results obtained can be hard to correlate with values from other evaluations (Reiter and Belz, 2009).

3. Automatic metrics evaluation was developed as an understudy to evaluations made by humans, based on the restrictions that such evaluations showed. These kinds of metrics evaluate the 'distance' between generated texts and a human-written reference text. They are considered cheaper than those that involve human evaluators. Their development is rapid if a corpus is available to be used for reference. Its repeatability is a quality to be considered. Nevertheless, its use is controversial, since it only measures 'how well the text is written', whereas human evaluators measure the impact that evaluated texts have on them (Papineni et al., 2002, Reiter and Belz, 2009).

2.1.5.1 Common automatic metrics

Over the next paragraphs we present the most referred automatic metrics in literature – BLEU and Meteor. At the end, we present the alternative used when BLEU and Meteor proved not to be the most accurate automatic evaluation metrics.

BLEU

BLEU stands for BiLingual Evaluation Understudy (Papineni et al., 2002). Evaluation is often a bottleneck point in NLG development, since evaluation with humans is expensive and time consuming. This automatic metric was developed to provide NLG developers with a tool to quickly evaluate the quality of the texts produced. As mentioned above, it measures how close the evaluated text is to a reference text written by humans. This evaluation is done from the individual terms of the evaluated text (words, sentence chunks or complete sentences), comparing them to a high quality reference text. The score obtained is then extrapolated to the entire text. Factors like intelligibility or grammatical issues are not taken into consideration with this metric. The score is expressed with a decimal number from `0` to `1`. The higher the value, the higher the similarity to the reference text. Score 1, or very close to 1, is only possible when the generated text is identical to the reference text. Due to the way this test is done – 'n-gram' comparison – the evaluation attributes acceptable scores when the evaluated text is compared with the entire reference text, and gives worse scores when are compared. It has been noted that scores lose precision from uni-gram to 4-gram. The precision of BLEU scores becomes unreliable above 4-gram.

Meteor

Meteor stands for Metric for Evaluation of Translation with Explicit ORdering (Denkowski and Lavie, 2014, Denkowski and Lavie, 2011), and was developed at the CMU Language Technologies Institute. Its developers argue that this metric has a high correlation with human evaluators (Denkowski and Lavie, 2011). The most recent version – Meteor Universal – was designed to be used with any language, and adapts to the required language automatically.

It works like BLEU, applying the evaluation algorithm to each sentence from the generated text. The Meteor algorithm makes an alignment between every term of the evaluated sentence and the reference sentence. By 'alignment' it means the direct mapping from one uni-gram of evaluated sentence to one uni-gram of reference sentence. Correspondences can be created if both words are equal, synonyms or derived from the same word. If the evaluated sentence is a paraphrase of a valid sentence, correspondence is also created.

The Meteor score is the harmonic mean between the number of correct alignments and the number of possible alignments. This second term has more weight than the first. Developed to correct specific problems of the BLEU metric, the Meteor metric is concerned with individual sentences, whereas the BLEU only works well when the entire text is evaluated.

2.1.5.2 Other approaches

When the sample is small, the previous metrics have shown to be inaccurate. Recent works propose new approaches, based on linguistic information (Mairesse and Young, 2014, Felice, 2012). One of these approaches is the Confidence Estimation (CE). It estimates the confidence that the system has in the quality of the output rather than evaluating the translation itself. CE metrics rely mainly on parameters and information from the NLG system, including features from the source and target text, instead of using human references. Another approach is Quality Estimation (QE), which is the opposite of CE. The focus is on how good the translation is, rather than on how confident a particular system is about its output.

The combination of these two approaches can increase accuracy in the assessment of translated texts. To process either a CE or QE evaluation, it is fundamental to extract statistics from the translated texts. Felice (2012) presents a set of 147 of these statistics, naming them as 'features'. Each feature is processed counting some kind of evidence from the text.

2.2 Related Work and state-of-the-art

In this section we present related and state-of-the-art work in the two fields of research most directly related to our work: data-to-text systems, with a focus on the ones employing data driven approaches; and classic template-based that will be used in the system to be presented in Chapters 3 and 4.

2.2.1 Data-to-text systems

This section presents Data-to-Text systems, defined by Reiter (2007) as *systems that generate texts from non-linguistic input data, which is typically numerical.* The major difference between 'classic' NLG systems and Data-to-Text systems is that this last category must analyze and interpret their input data, whereas the former does not need to. Of course, they do also need to decide how to communicate the produced utterance linguistically. The choice of these kinds of systems is a natural one, since we need to convey information that is stored in a relational database to our users.

The next subsections describe representative systems developed in the past decade. The assessment of the state-of-the-art began with a first evaluation, carried out at the beginning of the work, and has been improved throughout its development. We conclude with some very recent evolutions, and a comparison between systems.

2.2.1.1 Representative systems

During our research[2], we came across some systems related to generation for the Portuguese language (Fonseca, 1993, Ribeiro, 1995, Soares, 2001, Mendes, 2004, Oliveira, 2012). However, in the sub-field in reference, to the best of our knowledge, we only found two systems, that are presented below.

For simplicity, the systems are presented in chronological order.

[2]The search combined exploring lists such as the ones available at `http://www.nlg-wiki.org/systems/NLG_Systems_Wiki` or `http://aclweb.org/aclwiki/index.php?title=SIGGEN` and queries to common online resources such as Google Scholar or IEEE Xplore Digital Library. The majority of systems found are for English language.

Pollen Forecast for Scotland (2006)

The Pollen Forecast for Scotland system (Turner et al., 2006) was aimed at reporting the predicted pollen concentration in the different regions of Scotland, using text. Its job consisted of two related sub-tasks: the prediction itself, and the translation of the numerical data to text. The translation task is based on a parallel corpus of 69 data-text pairs. Each pair corresponds to a single item of pollen concentration data and its corresponding forecast. All forecasts were written by two expert meteorologists. Figure 2.6 presents a sample of the meteorology data, the corresponding forecast, written by a human meteorologist, and a map highlighting the meteorology data, in the table presented.

ValidDate	AreaID	Value
27/06/2005	1 (North)	6
27/06/2005	2 (North West)	5
27/06/2005	3 (Central)	5
27/06/2005	4 (North East)	6
27/06/2005	5 (South West)	8
27/06/2005	6 (South East)	8

'Monday looks set to bring another day of relatively high pollen counts, with values up to a very high eight in the Central Belt. Further North, levels will be a little better at a moderate to high five to six. However, even at these lower levels it will probably be uncomfortable for Hay fever sufferers.'

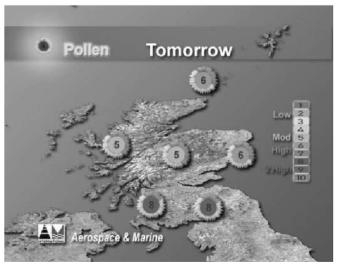


Figure 2.6: Sample of meteorology data; *left:* meteorology data; *center:* corresponding forecast; *right:* map highlighting the meteorology data, adapted from (Turner et al., 2006).

To be able to generate correct texts from the meteorological data, all human-written forecasts were analyzed and knowledge, about it, were extracted. Figure 2.7 presents a sample of analyzed message types, with corresponding data dependency and corpus coverage.

Message Type	**Data Dependency**	**Corpus Coverage**
Forecast	Pollen data for day of forecast	100%
Trend	Past/Future pollen forecasts	54%
Forecast Explanation	Weather forecast for day of forecast	35%
Hayfever	Pollen levels affect hay fever	23%
General	General Domain Knowledge	17%

Figure 2.7: Message categorization of the Pollen Corpus, retrieved from (Turner et al., 2006).

In addition, input data was analyzed in three steps: segmentation of the geographic regions by their non-spatial attributes (pollen values); segmentation of the geographic regions by their spatial attributes (geographic proximity); and, detection of trends in the general pollen level for the whole region over time. With these segmentation data sets, the Pollen Forecast system produced vectors of trends, for pollen prediction. These vectors were used to determine the correct forecast text to be produced.

This project upgraded to the SumTime (Sripada et al., 2003, Yu et al., 2003) system, whose authors have developed techniques for producing textual summaries of time series data.

Personage (2007)

Personage (Mairesse and Walker, 2007, Mairesse and Walker, 2010, Mairesse and Walker, 2011) stands for 'PERSONAlity GEnerator'. Its aim is to generate natural language utterances that respect the input provided, and the mood of the system's human user.

Personage was built respecting the entire architecture of NLG system defined by (Reiter and Dale, 2000), as can be seen in Figure 2.8. It can be classified as a data-to-text NLG system, since the main usage of Personage is used for assessing restaurant services in New York City. The goal is to classify restaurants and help users choose the restaurant that best suits their needs. The relevant data, which should be used as input, was stored in a database, with scalar values representing evaluative ratings for six attributes: food quality, service, cuisine, location, price, and atmosphere (Mairesse and Walker, 2011). Mairesse's approach assumes that the inputs to Personage were three. (1) One defining the high-level communicative goal, i.e. what information (recommendation or comparison) should be generated; second,(2) the data extracted from database, that is needed to achieve that goal, and (3) a set of generation parameters.

The first component is the 'content planner' where the structure of utterance to be produced is defined. After that, the sentence planner processes the previously generated content plan tree, selecting the syntactic templates for expressing individual propositions. The templates are then aggregated to produce the utterance's full syntactic structure. The pragmatic and lexical modules modify the last structure to introduce pragmatic effects and choose the correct lexeme, based on parameters provided by input data. The last stage is the 'surface realization' where all decisions taken, in the previous steps, are executed and a final utterance is produced. If the utterance needs to be converted to speech, it must be annotated with a prosody assigner, since Personage does not do that kind of work.

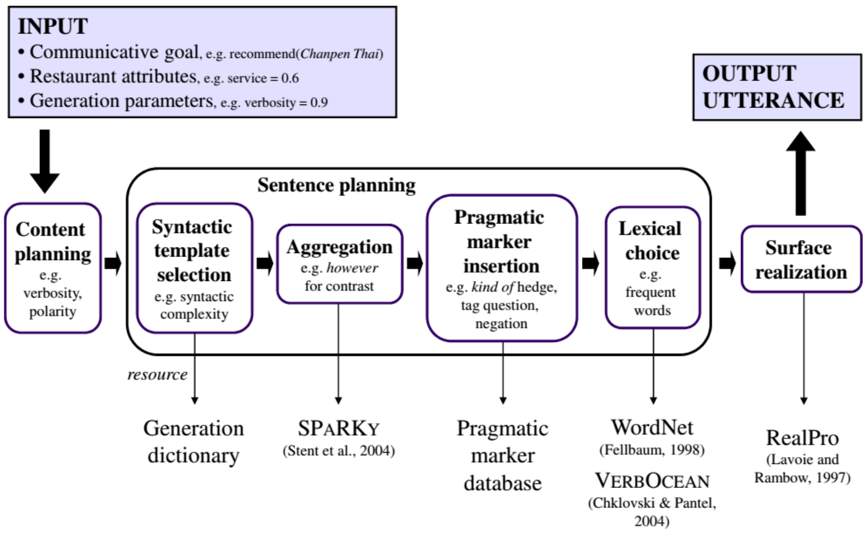

INPUT
• Communicative goal, e.g. recommend(Chanpen Thai)
• Restaurant attributes, e.g. service = 0.6
• Generation parameters, e.g. verbosity = 0.9
OUTPUT UTTERANCE
Sentence planning
Content planning e.g. verbosity, polarity
Syntactic template selection e.g. syntactic complexity
Aggregation e.g. however for contrast
Pragmatic marker insertion e.g. kind of hedge, tag question, negation
Lexical choice e.g. frequent words
Surface realization
resource
Generation dictionary
SPaRKy (Stent et al., 2004)
Pragmatic marker database
WordNet (Fellbaum, 1998)
VerbOcean (Chklovski & Pantel, 2004)
RealPro (Lavoie and Rambow, 1997)

Figure 2.8: PERSONAGE architecture. Retrieved from (Mairesse and Walker, 2011).

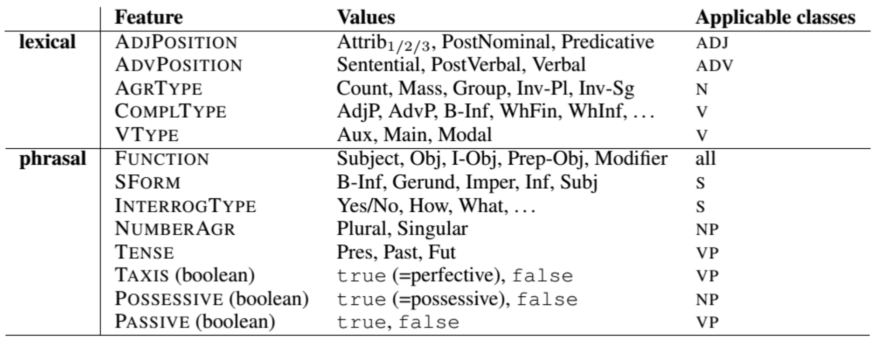
SimpleNLG (2009)

SimpleNLG (Gatt and Reiter, 2009, Reiter, 2016) is a Java API toolkit, developed under the supervision of Ehud Reiter at the University of Aberdeen – Scotland. It is an ongoing project. The most recent release is V4.4.8, from August 2016 (Reiter, 2016). SimpleNLG is a tool used to implement the last phase of Reiter's architecture onto NLG systems – the surface realization. It executes several tasks to achieve the final utterance:

1. define the constituents of the required utterance with lexical items;
2. assign features to constituents, such as those in the bottom panel of Figure 2.9;
3. combine constituents into larger structures;
4. the result from the last step is, then, supplied to a *lineariser*, where, before the final utterance is built, it is transformed based on features values.

One concern regarding SimpleNLG is its robustness. Structures that are not correctly formed do not generate an error. Instead, it will generate a general output, though a comprehensible one. Another characteristic is the separation between the morphological and syntactic operations. That characteristic is useful to lighten SimpleNLG's usage, when the final utterances only need to be correctly inflected.

SimpleNLG has been used successfully in several projects, both academic and commercial. It was developed to provide Reiter's teams a tool to create sentences in English only, and so it remains, which is a major drawback when the intention is to use this



	Feature	Values	Applicable classes
lexical	ADJPOSITION	Attrib$_{1/2/3}$, PostNominal, Predicative	ADJ
	ADVPOSITION	Sentential, PostVerbal, Verbal	ADV
	AGRTYPE	Count, Mass, Group, Inv-Pl, Inv-Sg	N
	COMPLTYPE	AdjP, AdvP, B-Inf, WhFin, WhInf, . . .	V
	VTYPE	Aux, Main, Modal	V
phrasal	FUNCTION	Subject, Obj, I-Obj, Prep-Obj, Modifier	all
	SFORM	B-Inf, Gerund, Imper, Inf, Subj	S
	INTERROGTYPE	Yes/No, How, What, . . .	S
	NUMBERAGR	Plural, Singular	NP
	TENSE	Pres, Past, Fut	VP
	TAXIS (boolean)	`true` (=perfective), `false`	VP
	POSSESSIVE (boolean)	`true` (=possessive), `false`	NP
	PASSIVE (boolean)	`true`, `false`	VP

Figure 2.9: Features and values available in SimpleNLG. Retrieved from (Gatt and Reiter, 2009).

tool in other languages. Several authors have been working on related tools, adapting SimpleNLG to other languages. So far, is possible to use SimpleNLG in French (Vaudry and Lapalme, 2013), in German (Bollmann, 2011), in Brazilian Portuguese (Oliveira and Sripada, 2014) and most recently, since 2016, in Italian (Mazzei et al., 2016). Despite being developed by different teams, the Brazilian Portuguese version (SimpleNLG-BP) can be considered a 'sibling' system to PortNLG, which is described further ahead.

BabyTalk (2009)

Work at a Neonatal Intensive Care Unit (NICU) can be very stressful for professionals working there (nurses and doctors), as well as for the families of the babies. At the start of their shifts, doctors and nurses need to absorb a large amount of data about the young patients, in a very short period of time. This data includes signals from sensors measuring physiological variables (e.g., heart rate, blood pressure, etc.), patient notes recording previous interventions, results of laboratory tests, and so forth. Authors argue that the correct absorption of such data is crucial to help professionals make the correct decisions for the babies' health. The main mode of presenting this data was, usually, the visualization of graphs and reports made by other professionals. This works well for experienced professionals, but junior professionals experience difficulties in absorbing all the data. One way of overcoming this handicap was to use a knowledge-based (expert) system which recommended specific interventions to the staff. As authors argue, this solution was not successful, since doctors tended to ignore its recommendations, particularly when they did not have an explanation as to why interventions should be carried out.

The BabyTalk (Portet et al., 2009, Hunter et al., 2011) project aims to join the two concepts: raw medical data and recommendations for specific actions aimed at medical staff. It aims to accomplish this by producing a text summary, in Natural Language, where data and instructions are put together. BabyTalk is a project that has five different sub-projects, each one related to a particular issue of NICU center activity:

- BT-Nurse (Hunter et al., 2011) was designed to automatically generate English summaries of electronically recorded patient data over a twelve hour nursing shift;

- BT-Doc will provide summaries of several hours of NICU data. This particular report is designed to be generated on demand. The period of time reported is bigger than others on the same project, hence it will probably need to use high-level abstractions of the data. Its target audience is young medical staff, to help them to make good decisions about interventions;

- Parents of NICU babies are a very important part in this equation. When their babies are hospitalized, they suffer from enormous stress. BT-Parent intends to generate summaries of NICU data for parents, to help them understand what is happening. Since parents have quite varied information needs, levels of medical expertise, and emotional states, BT-Parent intends to generate general reports, putting much more emphasis on user-modeling and adaptation than the systems intended for doctors and nurses;

- It is not only the parents of babies who need information about their health. Friends and extended family (e.g., grandparents) can offer support to parents, or even to the babies themselves. BT-Clan is intended to generate reports to this kind of users. For this purpose, it will probably need information about parents that is not available in current NICU databases;

- Last, but not least, BT-45 is intended to present reports that summarize 45 minutes of patient data. It is intended as a tool for nurses and doctors.

Out of all these sub-projects, BT-45 was the first to be implemented. BT-45 is a data-to-text system. However, instead of having only numerical data input, like weather reporting systems, BT-45's input is more heterogeneous. Figure 2.10 shows the system's architecture. Most data comes from a database where NICU data is stored. Another relevant aspect is the ontology, which is used both to represent domain knowledge and to support linguistic processing. BT-45 follows the data-to-text architecture suggested by (Reiter, 2007), hence the presence of Document Planning, Micro-Realization and Realization modules.

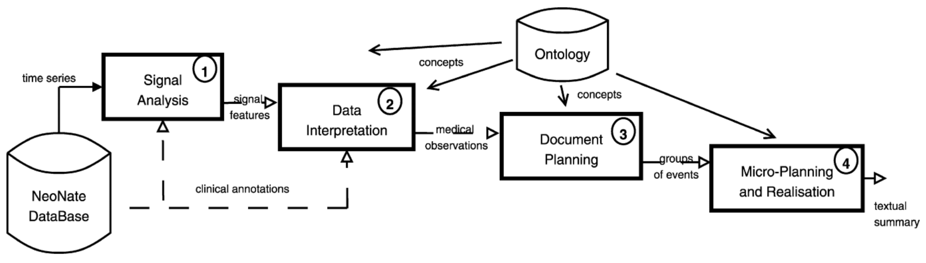


Figure 2.10: BabyTalk architecture (BT-45), retrieved from (Portet et al., 2009).

Mountain (2009)

Mountain (Langner and Black, 2009, Langner, 2010), *a **m**achine **t**ranslation approach for **n**atural **l**anguage **g**eneration*, was developed by Langner(2010), for his PhD. Mountain was the only one of the systems analyzed in this work to use the Moses tool (Koehn et al., 2007, Koehn, 2014). This statistical tool is used to train and process parallel aligned corpora. The translation model built by Moses is used by Mountain to generate an output sentence, translated from input language.

Mountain uses a scenario where tennis court users were asked to reserve it for their use, from one to several hours (e.g. one hour on Wednesday afternoon, Monday evening, etc.). The corpus collected the answers given. In order to produce an aligned corpus, each of the original 800 responses, were associated with a set of three tokens, expressing the requested scheduling. The first token expressed the court's availability, during the requested day (second token), and the period of the day – morning, afternoon or evening (third token). Example sentences include `111111 d2 t3` and `001110 d5 t1`. The collection of answers were classified as *output language*. The collection of corresponding tokens were classified as *input language*. Twenty young native speakers provided the answers to form the corpus. Then, the corpus was rated by a human evaluator. The majority of the corpus was considered good. Nevertheless, a significant portion of the corpus had a low score. The authors argue that this was likely due to errors made by the human responders.

An analysis of the corpus revealed that many answers given to one day were suited to other days. The authors realized that with a proper transformation, it would be possible to boost the corpus. This led to the expansion of the corpus from 800 sentences to nearly 4500 sentences. After the expansion, the corpus was trained with the Moses tools. First, all sentences were tokenized and case-normalized. Then, a 3-gram language model was created with the output language from the corpus. Finally, the entire corpus was used to produce a training model. The resulting training model was responsible for the translations carried out by Mountain. Figure 2.11 shows the training process from the initial parallel corpus to the trained models.

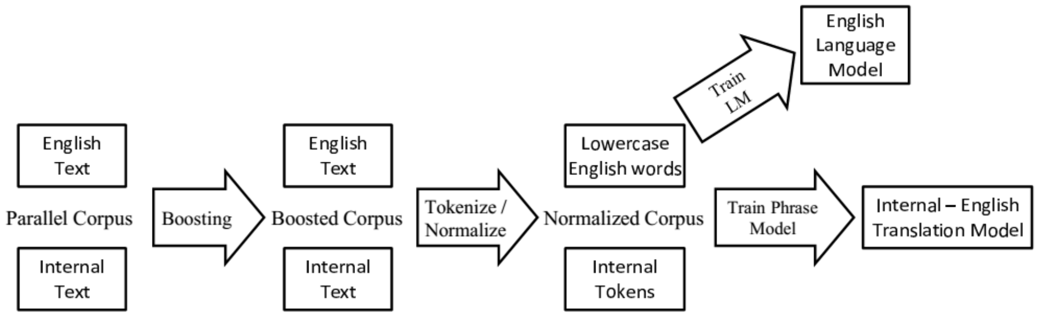


Figure 2.11: MOUNTAIN training process from initial parallel corpus to trained models,
retrieved from (Langner, 2010, p.70).

Figure 2.12 shows a sample of the output sentences produced by MOUNTAIN. Some sentences produced were directly translated from the *output corpus*, likely due the presence of a direct input–output pair in the corpus. Others were the combination of two or more original sentences. Overall, more than three quarters of the generated responses are not present in the original corpus. The assessment of the produced output was done by automatic metrics (BLEU and Meteor) and by human evaluators. As we will go over the results later, in Chapters 3 and 5, we need not discuss them in detail here. Nevertheless, MOUNTAIN's evaluations were compared to five other previous NLG systems (Langner et al., 2010), and conclusions pointed to a good result by MOUNTAIN.

000000 d5 t3	friday evening is completely closed
100000 d2 t2	the only time available is noon
111111 d4 t1	the court is open all morning
111111 d1 t3	you can reserve a court anytime on monday evening
100011 d5 t3	six , ten or eleven
010011 d3 t2	you can reserve a court at 1pm , 4pm and 5pm on wednesday
011001 d4 t3	any time but 6 , 9 and 10
111011 d7 d2	afternoon except the 3pm block
111100 d1 t2	you can reserve a court is free anytime from noon until 3
110111 d6 t3	saturday evening . ooh , that

Figure 2.12: Example of MOUNTAIN's output.
Left column: *input* sentence; right column: generated sentences,
retrieved from (Langner, 2010, p.71).

SINotas (2009)

To the best of our knowledge, SINotas (Novais et al., 2009, Araújo et al., 2010) is one of few data-to-text systems that uses the Portuguese language – in this case, Brazilian Portuguese. The goal of SINotas is to produce short textual reports on student's grades, weekly attendance rates and other academic information.

SINotas has a small corpus of 241 paired data-text records of students' academic performance data (Novais et al., 2009). All pairs were related to the students of a single professor. The same professor was responsible for writing the text that expressed the data's meaning. The use of a single professor, defined as a 'domain expert', was justified with the need for coherence on the meanings obtained from the raw data (e.g., students' grades) and semantics (i.e. the interpretation of the data according to a professor). The use of multiple 'domain experts' could generate contradictory descriptions of the same data. Figure 2.13 presents the 14 attributes to be processed by SINotas, their possible description and their possible values. Numbers correspond to the number of occurrences in the corpus.

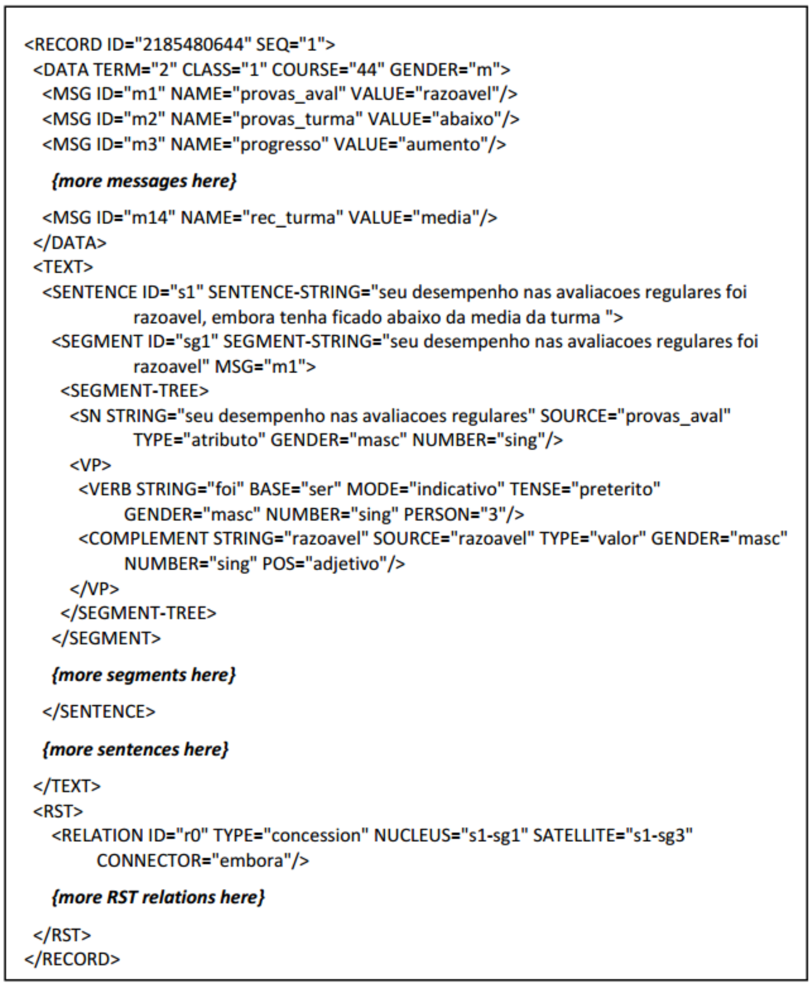
Attribute	Description	Possible values / number of instances
provas_aval	Regular exams grades	nao_realizou(50), muito_abaixo(30), razoável(40), bom_mas_baixo(6), bom(84), muito_bom(19), excelente(12)
provas_turma	Same, as compared to the entire class	nulo(50), abaixo(100), acima(91)
progresso	Overall progress throughout the term	nulo(50), declinio(50), menor_meio(48), maior_meio(65), aumento(28)
sub_aval*	Substitutive exams grades	nulo(223), muito_abaixo (16), abaixo(2), acima(0)
sub_turma*	Same, as compared to the entire class	nulo(214), abaixo(11), acima(16)
eps_aval	Practical exercises grades	nao_realizou(56), muito_abaixo(2), razoável(5), bom_mas_baixo(2), bom(22), muito_bom(33), excelente(121)
dev_ep1	Whether exercises were compulsory	nulo(207), sim(34)
freq_aval	Attendance to the lectures	nulo(188), nenhuma(44), insuficiente(9)
corel_nota_falta*	Lower grades attendance relation	nulo(215), sim(26)
mf_aval	Final term exams	muito_abaixo(81), razoável(41), bom_mas_baixo(5), bom(70), muito_bom(27), excelente(17)
mf_turma	Same, as compared to the entire class	nulo (58), abaixo(48), acima(135)
rec_aval	Recuperation exams grades	nulo(200), muito_abaixo(17), razoável(8), bom_mas_baixo(0), bom(16), muito_bom(0), excelente(0)
aband_rec*	Abandoned recuperation exams	nulo(235), sim(6)
rec_turma	Same, as compared to the entire class	nulo (204), abaixo(16), media(2), acima(19)

Figure 2.13: Attributes and possible values on SINotas, retrieved from (Novais et al., 2009).

Each one of the 241 paired data-text utterances, which form the corpus of SINotas, was represented on an XML data file, with three elements. <DATA> is the first element, where all the tokens presented in Figure 2.13 are represented, and for each one, a possible value of its universe; in the <DATA> element the possible text messages to be used on the abstracts sent to users are stored, related with tokens data from the

<DATA> element. Information about the target document, and the part-of-speech tags, which describe the sentences are also stored there. The final element is named <RST>, where the rhetorical linkers between sentences are stored. Figure 2.14 shows an extract of a SINotas record.

Despite being classified as Data-to-Text by its authors, SINotas can be considered a *templates* system, too, since all possible variations of produced texts were predicted, and well determined, in their application.



```
<RECORD ID="2185480644" SEQ="1">
 <DATA TERM="2" CLASS="1" COURSE="44" GENDER="m">
  <MSG ID="m1" NAME="provas_aval" VALUE="razoavel"/>
  <MSG ID="m2" NAME="provas_turma" VALUE="abaixo"/>
  <MSG ID="m3" NAME="progresso" VALUE="aumento"/>

   {more messages here}

  <MSG ID="m14" NAME="rec_turma" VALUE="media"/>
 </DATA>
 <TEXT>
  <SENTENCE ID="s1" SENTENCE-STRING="seu desempenho nas avaliacoes regulares foi
            razoavel, embora tenha ficado abaixo da media da turma ">
   <SEGMENT ID="sg1" SEGMENT-STRING="seu desempenho nas avaliacoes regulares foi
            razoavel" MSG="m1">
    <SEGMENT-TREE>
     <SN STRING="seu desempenho nas avaliacoes regulares" SOURCE="provas_aval"
            TYPE="atributo" GENDER="masc" NUMBER="sing"/>
     <VP>
      <VERB STRING="foi" BASE="ser" MODE="indicativo" TENSE="preterito"
           GENDER="masc" NUMBER="sing" PERSON="3"/>
      <COMPLEMENT STRING="razoavel" SOURCE="razoavel" TYPE="valor" GENDER="masc"
           NUMBER="sing" POS="adjetivo"/>
     </VP>
    </SEGMENT-TREE>
   </SEGMENT>

   {more segments here}

  </SENTENCE>

  {more sentences here}

 </TEXT>
 <RST>
   <RELATION ID="r0" TYPE="concession" NUCLEUS="s1-sg1" SATELLITE="s1-sg3"
        CONNECTOR="embora"/>

   {more RST relations here}

 </RST>
</RECORD>
```

Figure 2.14: Structure of a SINotas record, adapted from (Novais et al., 2009).

All texts that describe raw data are up to five sentences long, offering to students a description of their grades in relation to other students' grades. Figure 2.15 shows a sample text report, rendered in English since the original was written in Portuguese.

Your performance in the regular exam was good, and also above the average for your class.
Your grades had experienced an increase in the middle of the term, but fell again towards the end.
Your performance in the substitutive exam was slightly below the average for your class.
On the other hand, the results of your practical assignment were excellent.
Your final results were excellent and also above the average - congratulations!

Figure 2.15: A sample text report from SINotas, retrieved from (Araújo et al., 2010).

PortNLG (2013)

PortNLG (Silva Junior et al., 2013) is a recent tool, that processes the last phase of the NLG pipeline defined by (Reiter and Dale, 2000) – the Surface Realization, in the Portuguese language. PortNLG is a Java library, developed to be integrated with data-to-text systems, hence it is not in itself a real NLG system. It works similarly to SimpleNLG (Gatt and Reiter, 2009). Like SimpleNLG, it uses grammar rules, but because Portuguese grammar is more complex than English grammar, PortNLG needs an extra component – a lexicon module. The lexicon module is responsible, among other tasks, for giving technical information about Portuguese words.

Figure 2.16 presents a simple overview of PortNLG's architecture. Its input can be made in two distinct ways. One is like SimpleNLG –the application where PortNLG is integrated–, executes a sequence of calls to PortNLG methods, constructing, in this way, the desired sentence. Another, as presented at the Surface Realisation Shared Task (Belz et al., 2011) competition, where PortNLG was first presented, uses a single entrance, on a tree format, where the desired sentence is specified. Both methods use the same tools internally, and automatically produce an utterance.

Recent developments in Data-to-Text

We believe that Data-to-Text, or if preferred, Data2Text, systems are still a relevant issue in the context of NLG systems. The constant appearance of papers at relevant conferences and in journals is a proof of that. Over the next paragraphs, we present some of the most recent papers on the subject.

The interpretation of data charts is not always a straightforward issue. Aulia and Barmawi (2015) presents the Health Surveillance Chart Interpreter System (HS-

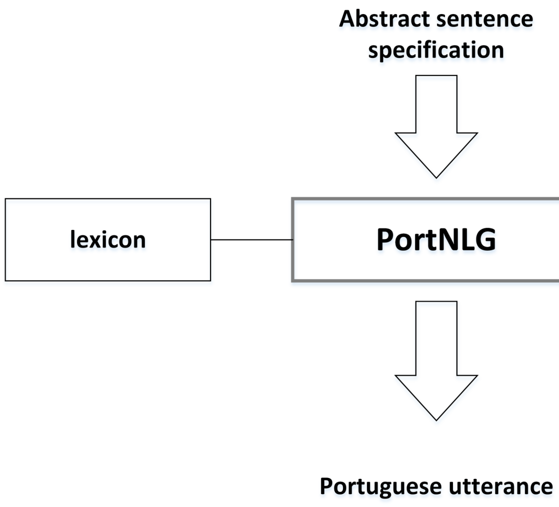


Figure 2.16: PortNLG system overview, adapted from (Silva Junior et al., 2013).

CISys), which aims to help Indonesian people interpret data charts. Since many people, particularly in rural areas, do not know how to do that, HS-CISys generates text summaries expressing the data on charts. Figure 2.17 presents the design concept of HS-CISys. Charts must be a line-chart, with one or two lines. HS-CISys is organized according to Reiter's Data-to-Text architecture. Sentences are produced by a set of templates, after the definition of what data should be sent to the user.

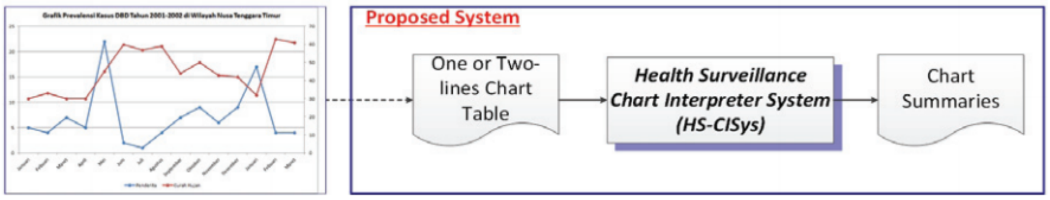

Proposed System
One or Two-lines Chart Table
Health Surveillance Chart Interpreter System (HS-CISys)
Chart Summaries

Figure 2.17: Concept Design of HS-CISys, retrieved from (Aulia and Barmawi, 2015).

Ramos-Soto et al. (2015, 2016) present a survey describing the evolution of NLG systems, with a special focus on Data-to-Text systems. They also describe the use of fuzzy logic to process the 'linguistic description of data' (LDD) – which is the task of extracting relevant information from some input data by producing an abstraction composed of linguistic terms (Ramos-Soto et al., 2016). It is characterized by their input data, the linguistic variables used, the fuzzy quantifiers, and evaluation criteria – and linked with a Data-to-Text approach.

Latin languages, as mentioned before, have specific traits that present some challenges which are difficult to overcome. Mazzei, Battaglino and Bosco (2016) proposed an adaptation of SimpleNLG to the Italian language. This adaptation, based on the French version of SimpleNLG, intends to overcome the problems with grammar and lexicon, evident in the base version of SimpleNLG, when applied to languages which are more complex than English.

Traffic congestion is still one of most challenging problems that a driver faces, especially in crowded cities. The system defined by Tran and Popowich (2016) aims to help reduce the frustration felt by drivers. Figure 2.18 shows the structure of this system. First, the GPS location of driver is collected, identifying the road they are using. Then, using that data, relevant traffic information (usually, concerning traffic incidents) is retrieved. After a generation module filters the information, a notification is produced and sent to the driver. Notifications can be delivered verbally, through a text-to-speech module, or by text, through some kind of text visualization device.

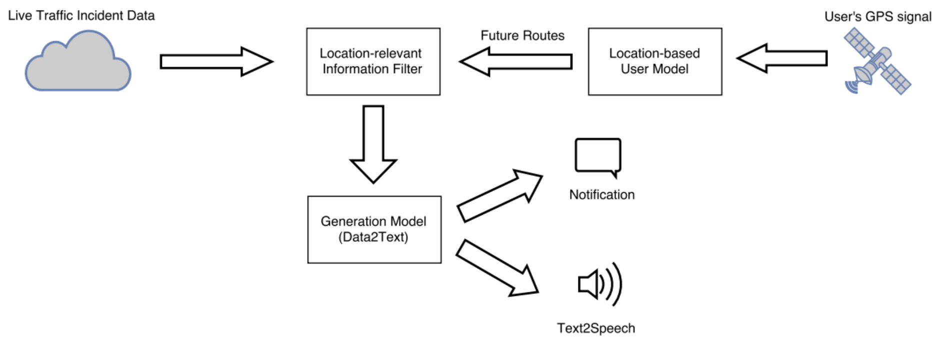

Live Traffic Incident Data
Location-relevant Information Filter
Future Routes
Location-based User Model
User's GPS signal
Notification
Generation Model (Data2Text)
Text2Speech

Figure 2.18: Structure of traffic notification system, that notifies location-relevant traffic information for road users, retrieved from (Tran and Popowich, 2016).

Systems related to diseases in human patients are one of the favorite themes for NLG systems. Alemzadeh and Devarakonda (2017) present a decision system, that relies on IBM Watson Patient Record NLP analytics to perform an extraction of patient's data, from structured data (data expressed on files, from database records) and unstructured data (human-written texts describing a patient's evolution over successive visits). This allows the extraction of features in order to classify the evolution of a patient's illness: 'Unknown', 'Controlled' or 'Not Controlled'. The system was tested and evaluated with data from patients with high blood pressure (hypertension), which according to the authors affects 29% of adults in the USA. Figure 2.19 shows the structure of this system.

2.2.1.2 Systems Comparison

Table 2.1 presents a comparison between the Data-to-Text systems analyzed above. Items like their name, year of release, language or type of approach are reported. It is clear that: (1) there are no prevailing domains, in the surveyed works. This is evidence of the versatility of NLG systems; (2) systems are mostly in English; (3) systems do not

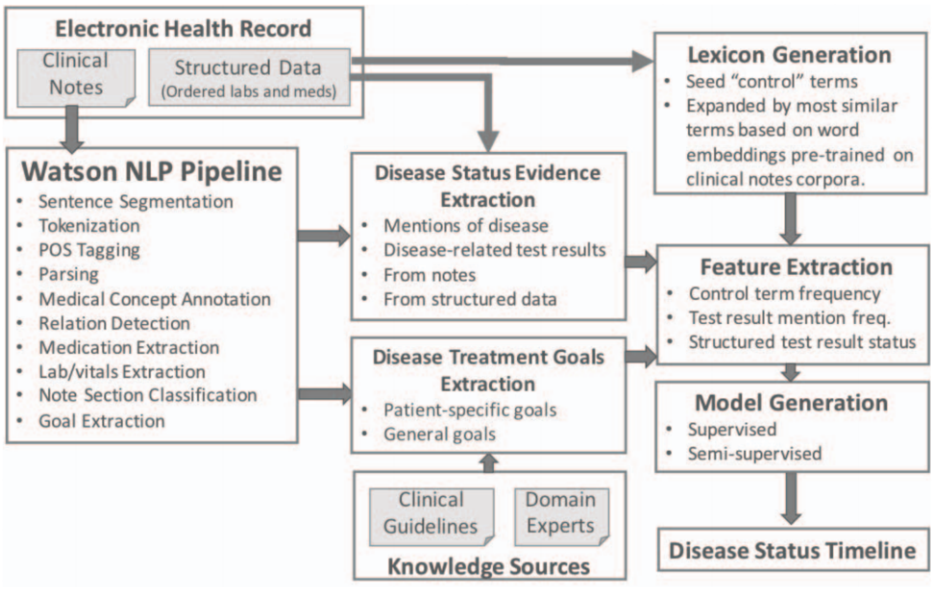


Figure 2.19: Overall system structure about extraction of data from Electronic Health Records, retrieved from (Alemzadeh and Devarakonda, 2017).

use the same approach to process the generation of sentences; (4) neither do they use a single approach exclusively. It is interesting to observe that some systems combine multiple approaches to achieve their goals. There are no prevalence of data-driven over classic systems, but, instead, a coexistence; (5) the emergence of new systems in recent years, is evidence that NGL systems, and in particular Data-to-Text, are a relevant topic for research.

2.2.2 Template-based NLG systems

Template systems are the oldest NLG systems that we came across. Nevertheless, they are still relevant, and several new systems are using them. In the next subsections, we present some of most significant systems using templates. For emphasis, we decided to group all template systems that use XML as their main engine at the end of this section. In order to simplify the presentation, systems are in chronological order.

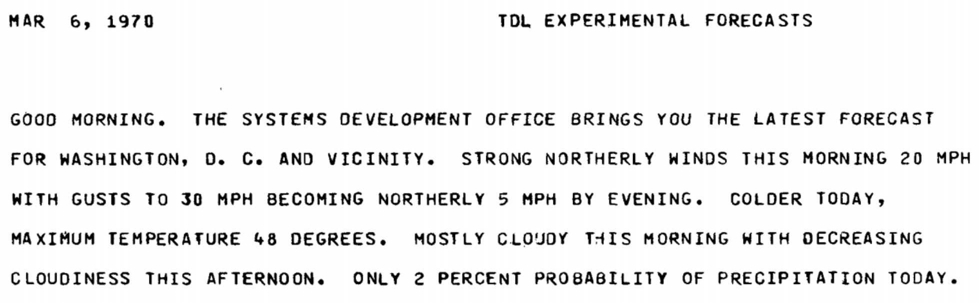
IFPS (1970)

Weather forecasts are probably the most common use for NLG systems. IFPS (Glahn, 1970a, Glahn, 1970b) was a system developed at the ESSA Weather Bureau (USA), which was able to produce weather forecasts, based on data processed by a mathematical equation. Automatic forecasts take into consideration the maximum temperature, the probability of precipitation or surface wind direction, that were

Table 2.1: Comparison of representative Data-to-Text systems and related tools

System	Date	Data2text	Domain	Approach	Language(s)
Pollen Forecast	2006	yes	forecast	classic & templates	EN
Personage	2007	yes	restaurants evaluation	data-driven	EN
SimpleNLG	2009	no	NLG tool	classic	EN
BabyTalk	2009	yes	babies health reports	classic	EN
Mountain	2009	yes	tennis court	data-driven	EN
SINotas	2009	yes	grades description	classic	PT
PortNLG	2013	no	NLG tool	classic	PT
HS-CISys	2015	yes	charts description	classic & templates	ID, IN
SimpleNLG-IT	2016	no	NLG tool	classic	IT
Traffic congestion	2016	yes	traffic congestion reports	data-driven	EN
System for Disease Status Identification	2017	yes	disease evolution identification	classic	EN

considered to be the most relevant forecast variables for most people. The weather predictions, of which an excerpt is shown in Figure 2.20, were produced by a computer program, from a set of about 80 phrases and sentences. To provide some variety, headings were chosen randomly from a set of possible headings.


MAR 6, 1970 TDL EXPERIMENTAL FORECASTS

GOOD MORNING. THE SYSTEMS DEVELOPMENT OFFICE BRINGS YOU THE LATEST FORECAST FOR WASHINGTON, D. C. AND VICINITY. STRONG NORTHERLY WINDS THIS MORNING 20 MPH WITH GUSTS TO 30 MPH BECOMING NORTHERLY 5 MPH BY EVENING. COLDER TODAY, MAXIMUM TEMPERATURE 48 DEGREES. MOSTLY CLOUDY THIS MORNING WITH DECREASING CLOUDINESS THIS AFTERNOON. ONLY 2 PERCENT PROBABILITY OF PRECIPITATION TODAY.

Figure 2.20: Excerpt of a weather forecast from IFPS system. Retrieved from (Glahn, 1970a).

The IFPS weather forecast report generator was replaced by the ICWF (Interactive Computer Worded Forecast) systems (Ruth and Peroutka, 1993).

TG/2 (1995)

The TG/2 system (Busemann, 1996) specializes in surface realization. As a result, it was designed to be integrated, as an external tool, onto multiple other systems. TG/2 can be used with canned texts, templates or context-free rules into a single formalism. Its output can be represented as simple text or as tabular output. It has the ability to re-use previously generated strings for additional solutions, and can be parameterized according to linguistic properties (grammar, fine-grained rhetoric, etc.). According to the authors, TG/2 has been used for more than 10 years (Busemann, 2005), and has evolved from a simple 'template' generator into a framework which processes simple templates as well as more complex tasks, with the development of small grammars. As an example, TG/2 findings were used in the development of the XtraGen system (Stenzhorn, 2002), which is presented in the XML Templates section, later on in this chapter.

YAG (1999)

YAG (Channarukul, 1999, McRoy et al., 2000) stands for Yet Another Generator. It is a template-based natural language generator for real time systems. It was designed to allow developers to quickly create systems without needing to write all output strings, and even, if they do not know or prefer not to specify all the grammar rules of the target language. YAG was developed in LISP and Java to allow its use on multiple platforms.

DEXTOR (2011)

DEXTOR (Narayan et al., 2011) was developed to provide natural language interaction between human and non-human players of interactive games. It introduced the notion of *typed-templates*, which are an enhanced version of templates. Figure 2.21 illustrates the use of a t-template, where an utterance is produced to be supplied to a player from a non-human player. T-templates consist of four components: utterances, parameters, return types, and parameter configurations. *Utterances* define text generated by templates. *Parameters* are placeholders for gaps in utterances. *Return types* are author-defined strings that tag the semantic meaning of a template. Each template must have, at least, one return type, that can be a general value, for instance, NOUN, or a specialized one, such as WEAPON. A *parameter configuration* is a mapping from t-template parameters to return types.

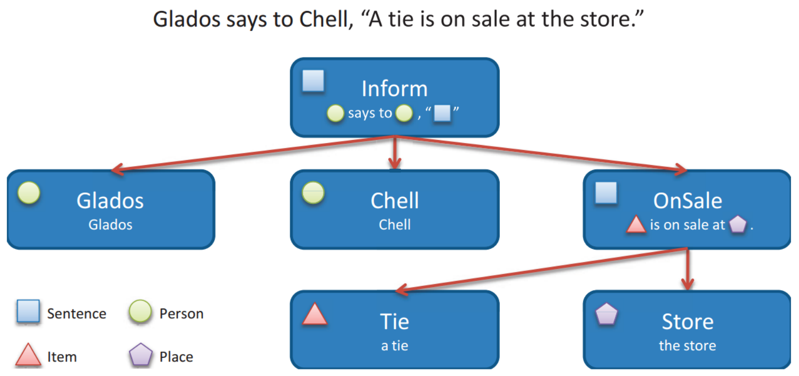


Figure 2.21: DEXTOR – sample t-template utterance production. Shapes in the top-left of a box denote return types. Shapes within an utterance denote a sample parameter configuration. Retrieved from (Narayan et al., 2011).

Crowdsourcing Language Generation Templates for Dialogue Systems (2014)

Online crowdsourcing can be a quick way to achieve a goal on a project, using the collaboration of many people. Most of people involved are unknown, which can lead to quality control issues. The Crowdsourcing Language Generation Templates for Dialogue Systems, proposed by (Mitchell et al., 2014), are a tool to help developers of dialogue systems build and extend corpora for their systems.

Figure 2.22 shows the pipeline of this system. It assumes that an initial set of manually executed and evaluated templates exists. Each of these templates has slots that are meant to be filled with data provided by the system. This first set is used as a *templates seed*. The first task is to collect a corpus of dialogues using the seed templates.

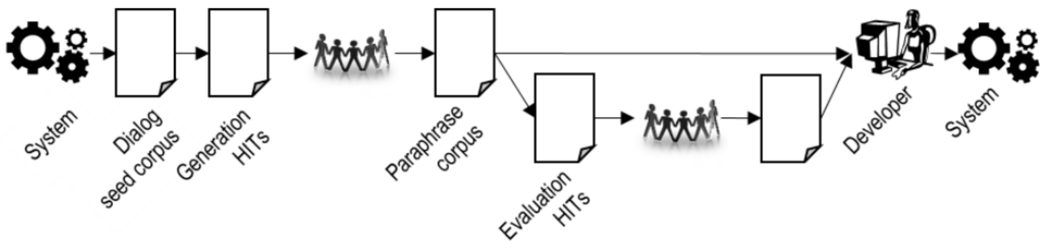


Figure 2.22: Pipeline for crowd-based development of natural language generation templates, retrieved from (Mitchell et al., 2014)

After that, a collection of HITs is produced based on the sentences obtained in task one. Each sentence is presented to a crowd-worker with a set of different options, based on several contexts. Using the options, each worker has to write a new sentence, respecting the meaning and coherence of the original sentence. The worker has to respect the original slots, if marked. This produces a new corpus based on crowd templates. It is up to system developer to decide which of these new templates should be added to the system.

At the same time, a set of evaluation HITs is constructed, and, again, presented to crowd-workers. This allows the system developer to assess the naturalness and suitability of the crowd templates. To be accepted, a crowd template must respect the original meaning of the original template, and sentences produced by it must sound natural in any system where the original templates were used.

Template-based Abstractive Meeting Summarization Framework (2014)

The Template-based Abstractive Meeting Summarization Framework (Oya et al., 2014) is a system aimed at summarizing conversations from a meeting. Instead of relying only on annotated data or using a collection of fused human utterances, which may result in grammatical mistakes, this framework uses another approach, which is shown in Figure 2.23.

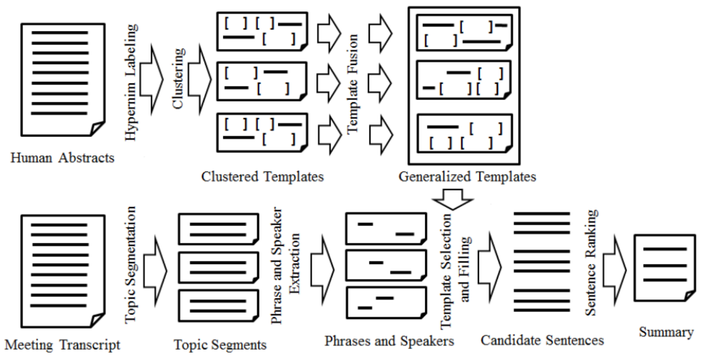


Figure 2.23: The Template-based Abstractive Meeting Summarization Framework.
Top: off-line Template generation module.
Bottom: on-line Summary Generation module.
Adapted from (Oya et al., 2014).

The main concern is the readability of the output text, which must be free of grammatical mistakes, and contain the relevant information from conversations. Two modules are used: an off-line template generation module and an on-line summary generation module. The first one creates templates from summaries produced by humans. The second one generates the text that is going to be presented to framework users.

The off-line module produces templates that are going to be used by the on-line module. These templates must satisfy two difficult requirements. They must be so specific that they should only accept relevant data and, on the other hand, they should be generic enough to be used several times. The first task, hypernym labeling, is done by classifying sentences which have subjects that correspond to meeting participants and contain active root verbs. This task uses the Standford Parser to parse sentences from the corpus with that information. In the following task, clustering, the templates produced by the first task are grouped by similarity. This action involves the creation of graphs, where each node is a root verb, and each edge represents a score denoting how similar the two words are. In the last task, template fusion, the clustered templates are generalized by applying a graph algorithm.

The on-line module, divided in four tasks, is responsible for generating the summary of the meeting conversation. The first task, topic segmentation, identifies the relevant information discussed in the meeting, making segments from it. These segments are used in the second task, phrase and speaker extraction. Here, all noun phrases present in each phrase of each segment are extracted, classified and scored. This allows the template to determine which sentences are relevant. The third task, template selection and filling, has the responsibility of selecting the best templates made by the off-line module, and fill them with the data processed in the first and second tasks. Finally, the last task, sentence ranking, evaluates the quality of the sentences produced, in terms of fluency, coverage of important topics and nature of the meeting. This evaluation produces a ranking of sentences. The best ranked sentences will be used in the summary.

2.2.3 Template-base systems using XML

The use of XML[3], and related technologies such as XSLT[4], is not a novelty in NLG development. In the following sections, several systems which use XML technologies are presented.

Exemplars (1998)

Exemplars (White and Caldwell, 1998b, White and Caldwell, 1998a) is an object-oriented, rule-based tool designed to support practical, dynamic text generation. The framework was used to develop a tool that generates reports, in HTML, to monitor the status of a project. Each report is created using fluent natural language, describing task progress, staffing, labor expenditures, costs or other issues related to the project.

[3] `https://www.w3.org/XML/`

[4] `https://www.w3.org/Style/XSL/`

XML and Multilingual Document Authoring: Convergent Trends (2000)

Dymetman, Lux and Ranta (2000) argue that the typical approach to using XML on NLG systems relegates it to a mixture of *tags* and *surface* (the text between tags). As a result, the authors present a new approach, where surface is removed from XML documents, and only generated by rendering mechanisms. Their aim is to separate the structure of the document from the semantics data. To achieve that, they present two proposals. One based on a mathematical constructive type theory and another based on a specialization of Definitive Clause Grammars.

D2S (2001)

The D2S (Theune et al., 2001) is a data-to-speech tool, designed to support the creation of data-to-speech for different languages and domains. D2S is a hybrid system, in which some parts of the generation process are based on general, linguistic principles, whereas other generation tasks are carried out using less flexible, application-specific methods. One of D2S's qualities is the focus on improving the prosodic quality of the system's speech output.

XtraGen (2002)

The XtraGen system (Stenzhorn, 2002) is based on XML and Java. The choice to use XML was motivated by its ability to convey data in a way that is easy to read and understand by both humans and computer programs. Java was chosen for its ease of programming. Specifically, for its interface and available libraries and packages, and particularly the ones related to memory management and XML processing. Additionally, XML was used to store the parameters and values related to the templates. The XtraGen solution was influenced by the ideas expressed in the TG/2 (Busemann, 1996, Busemann, 2005) and YAG (Channarukul, 1999, McRoy et al., 2000) systems. Figure 2.24 shows two samples of XtraGen XML templates. Each template has four sections, which influence its behavior: conditions – describe the exact circumstances under which a certain template can be applied and its actions executed. It is possible to have simple-conditions and complex-conditions (combination of several other conditions) ; parameters – defines parameters to format the generation process. Each parameter can be assigned a weight and thus a priority; actions – if all conditions of a given template have been tested successfully, the actions defined here are executed; constraints – they format the behavior of *actions*.

```
<template id="String"
          category="String">
  <conditions>
    Condition*
  </conditions>
  <parameters>
    Parameter*
  </parameters>
  <actions>
    Action+
  </actions>
  <constraints>
    Constraint*
  </constraints>
</template>
```

```
<conditions>
  <or>
    <and>
      <condition type="equal">
        <get path="/recall"/>
        <value>95</value>
      </condition>
      <condition type="less">
        <get path="/accuracy"/>
        <value>90</value>
      </condition>
    </and>
    <not>
      <condition type="exist">
        <get path="/exception"/>
      </condition>
    </not>
  </or>
</conditions>
```

Figure 2.24: Overview of a XtraGen template (left) and an example of some complex interleaved conditions (right), adapted from (Jokinen and Wilcock, 2003, p.229).

NewInfo (2003)

The NewInfo (Jokinen and Wilcock, 2003, Wilcock, 2005) system is an XML templates based system that aims to generate bilingual responses in Finnish and English for a Helsinki bus timetable inquiry system. This system uses the same *pipeline* described by (Reiter and Dale, 2000, p.60), to organize and build its output. Figure 2.25 shows its architecture. NewInfo has three main modules. The 'input manager', responsible for receiving speech input and converting it to understandable instructions to be processed by the 'dialogue module' where the output is generated, and then delivered to the user, by the 'presentation manager'.

The approach used was to create templates, as text plan trees, with gaps to be filled with sentence chunks obtained from the text specification tree. With this structure, NewInfo uses template-based text planning, instead of template-based generation. The only task assigned to XML is to describe the structure of the text. Then, the text plan is passed through the various stages of the generation pipeline for further processing. All processing is carried out by XSLT templates. Figure 2.26 shows the NewInfo transformation process in order to provide an answer to a simple user question:

User: *Which bus goes to Malmi?*

System: *Number 74.*

First, the generation pipeline, through the dialogue manager, starts creating an *agenda* element, which has a set of concepts. Its values are obtained from a timetable database. The dialogue manager labels each concept as 'NewInfo' or 'Topic', using

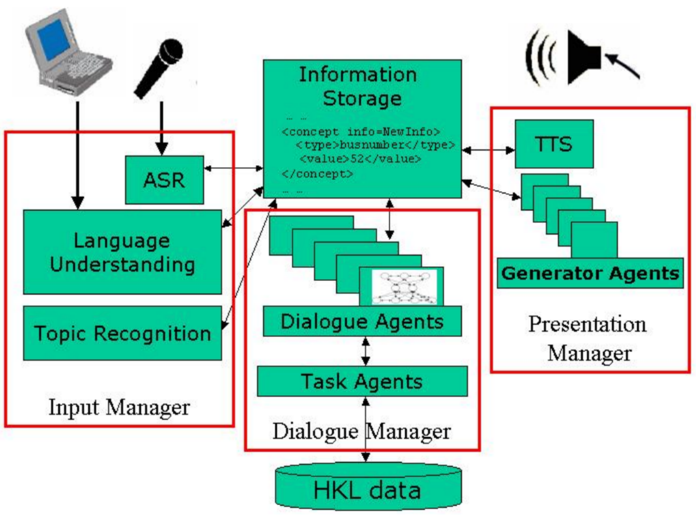


Figure 2.25: Adaptivity-oriented architecture of the *NewInfo* dialogue system. Retrieved from (Jokinen and Wilcock, 2003, p.229).

---

its knowledge of how the concepts relate to the current dialogue situation. The dialogue manager produces a *text plan*, describing the data that is relevant, from the 'agenda'. In this case, only the NewInfo data was used. The text plan was created by an XSLT transformation of the agenda. In the text planning stage, the concepts from the agenda are copied directly into the appropriate slots. The last step, again using the XSLT transformation over the text plan, is to create a text specification tree. The content of this last document was translated to Java Speech Markup Language and then communicated to the user through a speech synthesizer.

```
<agenda id="1">
  <concept info="Topic">
    <type>transportation</type>
    <value>bus</value>
  </concept>
  <concept info="Topic">
    <type>destination</type>
    <value>Malmi</value>
  </concept>
  <concept info="Topic">
    <type>bus</type>
    <value>exists</value>
  </concept>
  <concept info="NewInfo">
    <type>busnumber</type>
    <value>74</value>
  </concept>
</agenda>
```

```
<TextPlan id="1">
  <Message>
    <type>NumMsg</type>
    <concept info="NewInfo">
      <type>busnumber</type>
      <value>74</value>
    </concept>
  </Message>
</TextPlan>
```

```
<TextSpec id="1">
  <PhraseSpec>
    <subject cat="NP">
      <head>number</head>
      <attribute>74</attribute>
    </subject>
  </PhraseSpec>
</TextSpec>
```

```
<jsml lang="en">
  <div type="sent"> Number
    <sayas class="number">74</sayas> </div>
</jsml>
```

Figure 2.26: NewInfo XML elements – *left: agenda* element; *center top:* text plan; *right top:* text specification tree; *center bottom:* speech markup. Adapted from (Wilcock, 2005).

COMIC (2003) and FLIGHTS (2004)

Foster and White (2004) present several techniques for text planning with XSLT. They refer the capabilities of XML and XSLT to process and represent data. They describe its use with the COMIC (Os and Boves, 2003) and FLIGHTS (Moore et al., 2004) systems;

Text-SALSA XML (2004)

Erk et al. (2004) presents two XML formats, which were developed within the SALSA project (Erk et al., 2003) (which is related with the creation and annotation of a German corpus). Each XML is used to describe and encode semantics data in corpora. The TIGER/SALSA XML is the more general and extensive format, providing a modular representation for semantic roles and syntactic structure. The Text-SALSA XML is a lightweight version of the first one. It was developed with a special focus on human readability and ease of use.

Natural Language News Generation from Big Data (2015)

The definition of *big data* is not yet consensual. Nevertheless, it can be defined as a large amount of data, that traditional tools usually have great difficulty in dealing with (White, 2015, p.4–6). With that in mind, Haarmann and Sikorski (2015) proposes a system to generate newspaper news from big data sources. They argue that their system is useful since nowadays, increasingly, owners of large amounts of data have the need to generate reports, very often, for a limited number of readers. The cost of such a task is quite high. Here, the generation of text in natural language plays an important role, since it lowers the cost of producing texts related to what users want. The system was tested with data from football (soccer) matches, in German and Turkish. Reports were generated in German. In order for the system to understand the meaning of the available data, human annotation was needed. Taking into account that such databases are very complex, it is assumed that not all the data was fully annotated. Additionally, ontologies were used to acquire the semantic knowledge for the data. After that knowledge about the data was obtained, the system needs to know a) what kind of data is going to be included in a paragraph; b) in what order the paragraphs must be organized; and c) which data, effectively, needs to be grouped in a paragraph.

This system uses XML to convert relevant information provided by different sources in a uniform format. XML files are then used to express the structure of the templates. Figure 2.27 shows a sample of a semantic template. In this case, it references data from an outsourced font.

```
<mainClause mandatory="true"type="1"condition="$ENTSCHIEDEN">
<subject><phrase>$GEWINNER</phrase></subject>
<pblackicate tense="perfect">
<verb>gewinnen | siegen | sich durchsetzen</verb>
</pblackicate>
<completion optional="true"info="inStadion"/>
<completion info="Ergebnis.Gewinner, Ergebnis.GewinnerMit"/>
<extension>
<prepositionalPhrase>
<preposition>gegen</preposition>
<phrase>$VERLIERER</phrase>
</prepositionalPhrase>
</extension>
</mainClause>
```

Figure 2.27: Semantic template with reference to outsourced data, retrieved from (Haarmann and Sikorski, 2015).

2.3 Conclusions

In this chapter we started by describing the basics of human-computer interaction and why we need it. In particular, we discuss multimodal interaction, with a special focus on ‘output’. The use of Natural Language Generation (NLG) is described, and a brief description of its three approaches (templates, conventional NLG and trainable NLG) is made. A special interest was shown in a subtype of NLG systems – the data-to-text systems, where usually, data from a database is used as input.

The evaluation of the systems produced is a major concern. We describe two of the automatic metrics most used in this field, and introduce the use of ‘features’ to evaluate the quality of produced sentences.

We conclude by presenting a brief survey of state-of-the-art data-to-text and templates systems, where recent and relevant systems were described. The information presented in this chapter provides the background and views on the current state of this field, which are necessary to understand the options and development described in the next chapters, as well how generation results were assessed.

Chapter 3

A first translation-based specific domain Data-to-Text system for Portuguese

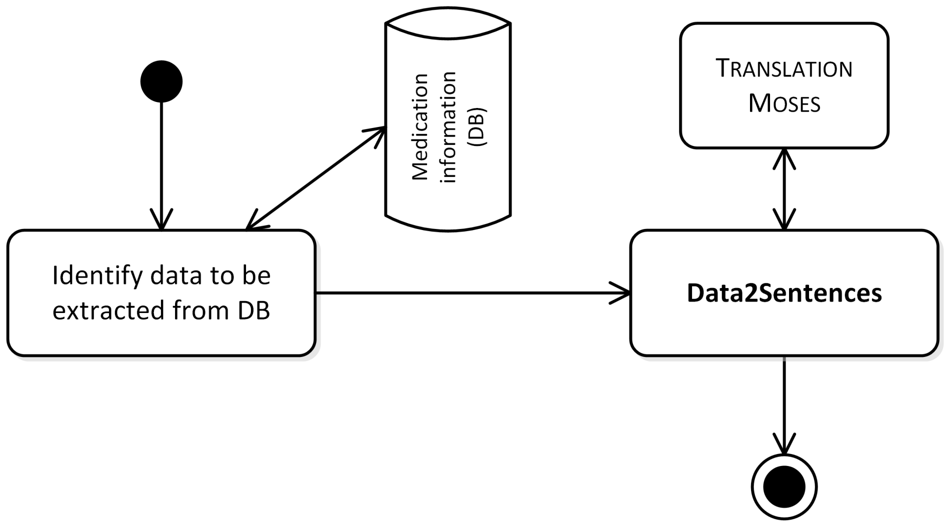
Driven by the increasing and ubiquitous need to convey information generated by sophisticated computer systems in Portuguese, this chapter presents a first experiment [1] in generating sentences from data in the Portuguese language. Medication plans were the chosen scenario. The adopted system is based on the use of machine translation and draws on recent work, such as Mountain. Two types of translation systems were developed and evaluated: one phrase-based and another using information about syntax.

3.1 System Overview – Medication2PT

The overall architecture of the first developed system, named Medication2PT, is shown in Figure 3.1. Its mission is to generate messages in natural language, related to the prescription of medicines, in order to be used as a component of a broader system, such as a Medication Assistant (Ferreira et al., 2014).

The central part of this system, and the subject of this chapter, is a module, named Data2Sentences, which is able to create sentences, in response to vectors with data, provided as input. If several input vectors are sequentially fed to the Data2Sentence module, an aggregation module can be used to join the several generated sentences to produce a small text.

[1] Published in (Pereira and Teixeira, 2015)


Medication information (DB)
Translation Moses
Identify data to be extracted from DB
Data2Sentences
Figure 3.1: Architecture of our first Data-to-Text system – Medication2PT

To achieve the goal of generating sentences based on information regarding a person's medication plan, additional modules are needed, to provide the input vectors to the Data2Sentence module (the Database module), and to make the conversion from input to output language (the Translation module).

The **Database** (DB) **module** is the component responsible for storing all data: the user's personal information, information on the medication, prescriptions, etc..

The **Translation module**, based on the Moses translation system, is responsible for the translation of the input vectors, sent by the Data2Sentence module, to Portuguese. To perform this task, first Moses must be trained with a corpus consisting of the two perfectly aligned languages. Every sentence in the input language must match a phrase in the output language. The input language consists of values by the DB module. Output language consists of the expressions in Portuguese we want Moses to generate.

The **Data2Sentences module** is responsible for receiving requests from users and interacting with the data stored in the Database. It is also responsible for sending messages in the input language to the Moses-based Translation module and receiving an answer in the output language. Finally, it is in charge of processing the answers and transmitting them to the application it is integrated in.

3.2 Creating the Translation Module

To create the core component, the TRANSLATION module, we need to define the input language, to create a parallel corpus with aligned examples of this language and sentences in Portuguese and train Moses with adequate models. These are the subjects of the following subsections.

3.2.1 Input language

The input language reflects the data that is obtained by a query to the database, which has been divided into 9 data types: name and surname of the user, courtesy, name of drug to take, drug type (ex: pills), method for intake, dosage to be taken and frequency. These terms were concatenated to the values of the primary keys corresponding to the records selected from the database. In this way, each phrase reflects the user's data and what he/she must do in terms of medication.

Correspondingly, in the output language we will have a sentence in Portuguese that expresses the information encoded in the input vector that is suitable to be read by human users.

Table 3.1 shows examples of these two languages.

3.2.2 Parallel Corpus

The corpus used in this first experience was obtained through the following process:

1. Fifteen people, aged 18 to 50 years old, were asked to write sentences related to the task of giving medication to a patient. 126 sentences were obtained.

2. Using the same strategy described in (Langner, 2010), the original corpus was expanded to 643 sentences.

3. This expansion was carried out because we realized that each original sentence, which applied to a specific person and a specific medicine, could also be applied to other people or medicines, if properly adjusted to the new context. To do that, the following steps were followed:

   (a) Keeping the original corpus sequence, every sentence was manually edited and words related to the corpus theme (medicine administration) were replaced with a specific *token*. Table 3.2 presents two samples of this step.

Table 3.1: Part of the parallel aligned corpus used in the development of the first data-to-text system. The first 10 lines of the training part of the corpus are presented. Input and output are presented side by side, being the input at the left column.

input	corresponding sentence
pessoa32n saudacao_0 pessoa0a medicamento21 tipo0 tomar0 cor00 dose0 freqtoma00	Helena pode tomar agora o Seretaide.
pessoa0n saudacao_0 pessoa0a medicamento14 tipo1 tomar2 cor00 dose4 freqtoma02	Vai-se deitar então tome quatro comprimidos Primperan.
pessoa40n saudacao_0 pessoa0a medicamento0 tipo1 tomar2 cor03 dose0 freqtoma04	Ao almoço toma o comprimido branco Leonardo.
pessoa0n saudacao_m pessoa12a medicamento0 tipo8 tomar3 cor00 dose0 freqtoma02	Antes de deitar senhor Lima não se esqueça da bomba de inalação.
pessoa17n saudacao_f pessoa0a medicamento0 tipo4 tomar2 cor04 dose0 freqtoma02	Antes de deitar faça a toma das gotas amarelas dona Cristina.
pessoa0n saudacao_0 pessoa0a medicamento3 tipo1 tomar2 cor10 dose4 freqtoma05	É hora de jantar tome os quatro comprimidos laranja do Ibuprofeno.
pessoa36n saudacao_0 pessoa0a medicamento19 tipo8 tomar3 cor00 dose0 freqtoma01	São horas de acordar e de colocar a bomba de inalação Nasomet daqui a três horas João terá de colocar de novo.
pessoa21n saudacao_f pessoa0a medicamento2 tipo4 tomar2 cor00 dose5 freqtoma01	Dona Elisabete assim que acordar deve tomar cinco gotas de Clorocil.
pessoa37n saudacao_0 pessoa0a medicamento23 tipo4 tomar2 cor00 dose4 freqtoma05	Está na hora de jantar Jorge não esqueça de tomar as quatro gotas de Guttalax.
pessoa78n saudacao_f pessoa0a medicamento3 tipo1 tomar2 cor00 dose3 freqtoma04	Dona Teresinha está na hora de almoço tome os três comprimidos Ibuprofeno.

Sentences are shown in Portuguese alongside a translation to English. Table 3.3 shows all the *tokens* used in this step.

(b) After this step, each of these new 126 sentences was randomly replicated, between 3 and 7 times. The original sequence was maintained. As mentioned, 643 new sentences were obtained.

(c) The next phase is the replacement of tokens by their correspondent values. A small database was made with possible values for each token. In this case, 80 first names, 33 last names and data for 28 medicines were defined. We

Table 3.2: Two sentences from original corpus and corresponding transformation with *tokens*.

Adriana podes tomar agora o Clorocil (pt) Adriana, you can take now the Clorocil (en)
NOME podes tomar agora o MEDICAMENTO (pt) NAME, you can take now the MEDICINE_NAME (en)
Sr.a Teresa antes de deitar coloque as gotas Clorocil (pt) Mrs Teresa before bedtime put the Clorocil drops (en)
FORMA_TRATAMENTO NOME antes de deitar coloque as TIPO MEDICAMENTO (pt) COURTESY NAME before bedtime put the MEDICINE_NAME MEDICINE_TYPE (en)

Table 3.3: *Tokens* and their correspondences.

Token	Correspondence
Name	first name of person who are going to take the medicine
Last_Name	last name of person who are going to take the medicine
Courtesy	salutation to whom are going to take the medicine (corresponds to Mr, Mrs, Miss, etc.)
Medicine	medicine name
Quantity	quantity of medicine to take
Medicine_Type	medicine type (pills, drops, etc.)
Time	hour of day when the medication should be taken
Color	medicine color

identified the name, type (drop, pill, etc.) and color of each medicine. We also defined 11 possible periods to take the medicine, and 6 possible quantities of medicine to take. Every token was randomly replaced with values obtained from the database. When data was related (e.g. medicine name, medicine type and color), choosing a value implies choosing the related data to maintain integrity.

(d) At the same time that we process each one of these new sentences, we collect all data assigned to each token. As a result, we obtained two files, with 643 perfectly aligned sentences. The file with the sentences is the *output* language, and the file with the token data is the *input* language of our corpus.

(e) The final stage was an analysis, performed by a single person, to the overall quality of each new sentence. Only grammar and syntax errors were corrected.

3.2.3 Corpus preparation for training and testing

To train our system and test it, our corpus was split in two disjointed sets. The *10-fold cross-validation* (Hall et al., 2011, Kohavi, 1995, Salzberg and Fayyad, 1997) technique was used. One set, for the test, has 10% of the sentences. The remaining 90% of sentences were used to train the system. Using this separation, we prepare 10 different groups. Because all the sentences from the *output* language were obtained by replication, if these 10 different groups were obtained by simple random selection, it could be the case that some sentences, with the same 'seed', have more influence than others in the same group. That is why we made the separation slightly differently:

1. Keeping the original sequence of sentences, each sentence was tagged with a letter from 'A' to 'J'. Each letter identifies a different group.

2. The sequence A–J was assigned sequentially, renewing itself continuously. This way, sentences with the same 'seed' are each in different groups.

3. After this classification, the entire corpus was reordered by tag name (letter from A to J).

4. This process ensures that the creation of the 10 different groups is random. Each 'seed' sentence was obtained independently. The replication of every one of the 126 'seed' sentences of the final corpus was done randomly 3 to 7 times. The replacement of tokens by their correspondent values was also done randomly. Every sentence in a group has a different seed.

After the creation of 10 different groups of corpus sentences, 10 sets for training and testing were made. We named each set, again, with letters from A to J. Each set has two files. One file, named with the letter name and the suffix `-test` contains all sentences that belong to the group named with the same letter. The other file, named with the same letter and with the suffix `-train` contains all sentences that belong to the other groups.

3.2.4 Some statistics

Some statistical data about the *output* language can be found in Table 3.4.

Table 3.4: Some statistical data about *output* language.

Number of sentences	643
Number of words	7212
Average number of words/sentence	11
Max. number of words/sentence	30
Min. number of words/sentence	4

3.2.5 Developed Systems

Two variants of the system were developed, taking advantage of the two main variants of automatic translation systems available in Moses: `phrase-based` and `tree-based` (using syntactic information).

Phrase-based system

To carry out this first experience, after collecting the corpus, its expansion and segmentation training was performed. The various procedures were executed as prescribed in Moses documentation (Moses, 2014). Moses has several scripts specially prepared for the execution of each phase. For each of the 10 train sets, a complete independent train and test was performed. Each set was subjected to the same procedures, carried out in the same order.

The first task consisted of the preparation of the corpus, starting by its tokenization. This operation consists of separating, based on the blank spaces, each of the elements that make up each of the phrases of the corpus. An `element` is a word or a punctuation mark. For the output language this procedure was performed using the `tokenizer.perl` script. For input, it was not necessary to carry out this task, as its method of creation already provides the tokens.

The cleaning phase was executed, but had no practical effects. In fact, this phase is intended to eliminate malformed or excessively long sentences (a practical limit of 80 words was empirically defined). In our corpus, all sentences are under this limit, well-formed and properly aligned.

The second task consisted of training the language model. At this stage, the intermediate tools that will assist Moses in the training of the translation system are created. This intermediate model is intended to ensure a fluent generation of text, and therefore it is done in the output language. Language models were created using IRSTLM (IRST, 2011).

In this step, the `n-gram` parameter is also set. By default, it has the value 3, which means that the template will make groupings of 3 words. These groups are then used in the creation of the output text. Essentially, three steps are followed. First, every line is prefixed with the term <`s`> and suffixed with the corresponding term <`/s`>. Then, the language model is built. Finally, the created language model is compiled.

The third task consists of training the translation system, using the language model created in the previous task and GIZA++ (Och, 2011). The training process creates, among others, the files `moses.ini` and `phase-table.gz`, required to configure and use MOSES.

The last task is to test and use the model trained to generate the desired texts.

Syntax-based System

A new experiment was carried out, using the same sets, creating a syntax-based system. The major difference, from the previous training, is the fact that the training for this model focuses on building a tree that represents the syntax of output. Subsequently, it is this tree that is used, instead of the words directly.

The main difficulty arose from the choice of the parser, since we need to classify each word of output sentences. Our main requirements for the parser were:

1. that it would be able to produce an output that could be easily adapted for use in Moses, while being compatible with the system's tools for manipulating trees;
2. that it would be possible to integrate the parser into our system;
3. that its use is free.

Considering these points, several parsers were installed and tested, namely: Palavras[2], Freeling[3], Tycho Brahe[4], TreeTagger[5], Turbo Semantic Parser[6] and Stanford Parser[7].

The choice fell on the last one, with the adaptation made by the LX-CENTER group (Language Resources and Technology for Portuguese), University of Lisbon, Portugal (Branco and Silva, 2004). Despite the limitations noted, it was nonetheless the parser which best corresponded to our requirements.

[2]http://beta.visl.sdu.dk/contact.html
[3]http://nlp.lsi.upc.edu/freeling/
[4]http://www.tycho.iel.unicamp.br/ tycho/apps/dbparser-files/
[5] http://www.ims.uni-stuttgart.de/projekte/corplex/TreeTagger/DecisionTreeTagger.html
[6]http://labs.priberam.com/Resources/TurboSemanticParser.aspx
[7]http://nlp.stanford.edu/software/lex-parser.shtml

3.3 Results

In this section, we present examples that represent the generation capacities of the systems which were developed, followed by a formal evaluation using common metrics and evaluation by humans. The possible influence of the way the corpus was divided (in the training and testing stages) is also discussed. We conclude this section with some information on the integration of the systems which were developed in a Medication Assistant for Smartphones.

3.3.1 Examples

Table 3.5 presents several selected examples in order to illustrate the various types of results obtained. The aim is to familiarize the reader with what was in fact obtained using both types of systems.

At the top of the table, numbered 1 to 3, are some examples of internal representation and the corresponding output generated by the phrase-based systems (specifically trained with the corpus). The generated sentences vary in quality, one of them being intelligible, another acceptable and the last one correct.

The second part of the table (numbers 4 to 7) presents the alignment between the phrases created by humans and those created by the system (regardless of type) for the same input. The phrases are presented in lowercase in order to highlight their differences, as explained below. For a certain input pertaining to our corpus, when a sentence is generated by our system it can be – and normally is – different from the sentence in the training corpus. These differences usually consist of: the addition of new words, erasing of words or a change in their order. We use "***" to highlight the addition or erasing of words and upper case to identify a change in word order. These occurrences arise only in the phrase generated. When we represent them in the sentence written by humans it is only to make the differences between the two sentences more obvious.

Examples 6 and 7 show generated phrases which are considered understandable and good in terms of naturalness, but which are completely different from those produced by humans (in our corpus). This type of sentences are a major challenge in the assessment stage, as they are usually considered errors by automatic evaluation metrics. These limitations in automatic evaluation made us reconsider our initial decision to only use automatic evaluation, and so an additional evaluation by humans was performed.

In the third part of the table the differences between the results obtained by the 2

Table 3.5: Selected examples of output for the two types of system.

Examples showing input and generated sentence	
#	Example
1	pessoa45n saudacao0 pessoa0a medicamento17 tipo0 tomar0 cor00 dose0 freqtoma00 Luís Pulmicort de tomar agora o
2	pessoa61n saudacaom pessoa0a medicamento4 tipo0 tomar0 cor00 dose2 freqtoma00 senhor Paulo tome dois comprimidos de Salazopirina
3	pessoa49n saudacao0 pessoa0a medicamento0 tipo2 tomar2 cor09 dose0 freqtoma10 Marcelo aplique ao tomar a cápsula branca e azul de dez horas em dez horas
System (S) output aligned with sentences by humans (H)	
4	H: dona denise assim que se levantar não se esqueça de tomar OS COMPRIMIDOS nicotibine S: dona denise assim que se levantar não se esqueça de tomar O COMPRIMIDO nicotibine
5	H: DEVE TOMAR AGORA ao acordar *** a bomba de inalação DE pulmicort AUGUSTO S:*** *** AUGUSTO ao acordar APLIQUE a bomba de inalação *** pulmicort ***
Outputs very different from human examples, but acceptable	
6	H: É HORA DE ALMOÇAR marcos não se esqueça de tomar *** quatro gotas de guttalax *** *** S:*** *** *** *** marcos não se esqueça de tomar AS quatro gotas de guttalax AO ALMOÇO
7	H: *** *** É MEIO-DIA TOME AS três gotas de zaditen *** PATRÍCIA S:PATRíCIA NÃO SE ESQUEÇA DE TOMAR três gotas de zaditen AO MEIO-DIA
Output by the 2 systems for the same input (Ph identifies phrase-based; S identifies system using syntax (tree-based))	
8	Ph: Patrícia não se esqueça de tomar três gotas de Zaditen ao meio-dia S: Patrícia ao Zaditen gotas tome de três meio-dia
9	Ph: Senhora Carvalho após o seu almoço tome cinco comprimidos de Duphaston S: Senhora Carvalho Duphaston comprimido branco cinco almoço

types of system are exemplified. For each of the subsets we used the same input vector in order to ensure the comparability of the sentences.

3.3.2 Comparative Evaluation of the two types of system

Method

Taking into account the main goal of having information on the absolute and relative performance of the two types of systems created, the process began by training 10 systems for each of the two system types. Default values were adopted for most of the parameters (example: the value of 3 for the n-gram).

Once trained, the systems were evaluated, first with the training corpus and, after checking the proper operation of the system, with the corresponding test set. The BLEU (Papineni et al., 2002) and Meteor (Denkowski and Lavie, 2014, Denkowski and Lavie, 2011) metrics were adopted for the evaluation, following the example of the evaluation of Mountain performed by Langner (Langner, 2010).

In addition, to complement BLEU and Meteor, an assessment was carried out by humans. The assessed phrases were chosen from among those generated by the 2 systems. All these sentences were obtained with the same set of the test part of the corpus – set F. After being chosen, sentences were ordered randomly and evaluated in terms of intelligibility/comprehensibility, sentence structure and overall quality. To simplify the evaluators' task, the responses to comprehensibility and structure were reduced to only 3 options. Concrete information about the questions and response options is shown in Table 3.6.

A total of 11 people participated in the evaluation. They had a broad age range (16 to 58 years old) and varied training and occupational backgrounds (including, for instance, students, administrative assistants and teachers).

Table 3.6: Information regarding the questions and possible answers used in the evaluation of the generated sentences by humans.

	Question	Possible answers
Intelligibility	Understandable ?	0 = No 1 = In part 2 = Yes
Structure	Structure of Sentence	0 = Bad (several problems) 1 = Fair 2 = Good
Quality	General Quality ?	from 1 to 5, where 1 = Bad 5 = Excellent

For syntax-based systems, an experiment was made on the effect of the weight assigned to the language model in the process of decoding. Several weights of the language model were tried (using the training corpus), having come to the conclusion that there was a very positive effect on the BLEU and Meteor metrics when this weight was significantly higher than the default value. In view of this result, this new value (10) was adopted for the evaluations of all syntax-based system with the test set.

Results of the automatic evaluation

The results obtained, in terms of BLEU and Meteor metrics for both types of systems, are shown in Figures 3.2 and 3.3, using means and 95% confidence intervals for the mean.

In terms of BLEU, in Figure 3.2, the phrase-based system gets a better average performance in all parameters, except for the Bleu1 –the parameter related to `unigrams`– performance being significantly better for the global parameter BLEU and Bleu2 (noticeable in the figure by the non-overlap of confidence intervals). The best value of BLEU obtained was 0.245.

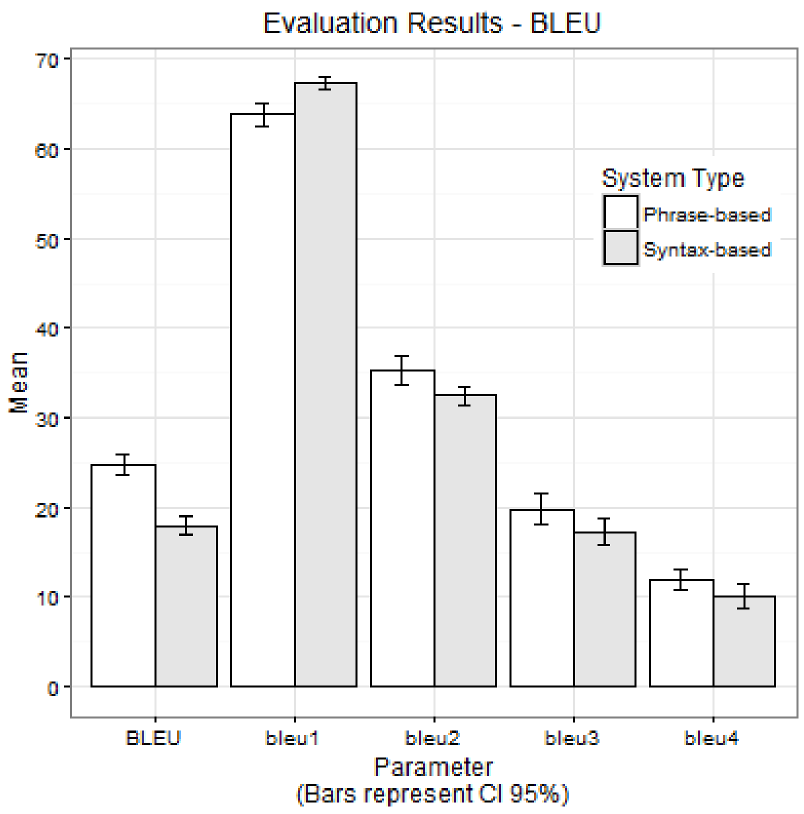


Figure 3.2: Results for the BLEU-based evaluation.

The previous results are generally confirmed by Meteor (see Figure 3.3). In this case, all parameters are worse for the system based on syntax – the differences in terms of Recall, Penalty due to Fragmentation and Final Score are particularly noticeable. While both systems are capable of a similar performance in terms of precision, generally selecting the correct words for the phrase, the syntax-based system is more fallible on the inclusion of words that should be part of the sentence (lower recall).

The results obtained by the systems based on syntax are certainly related to the

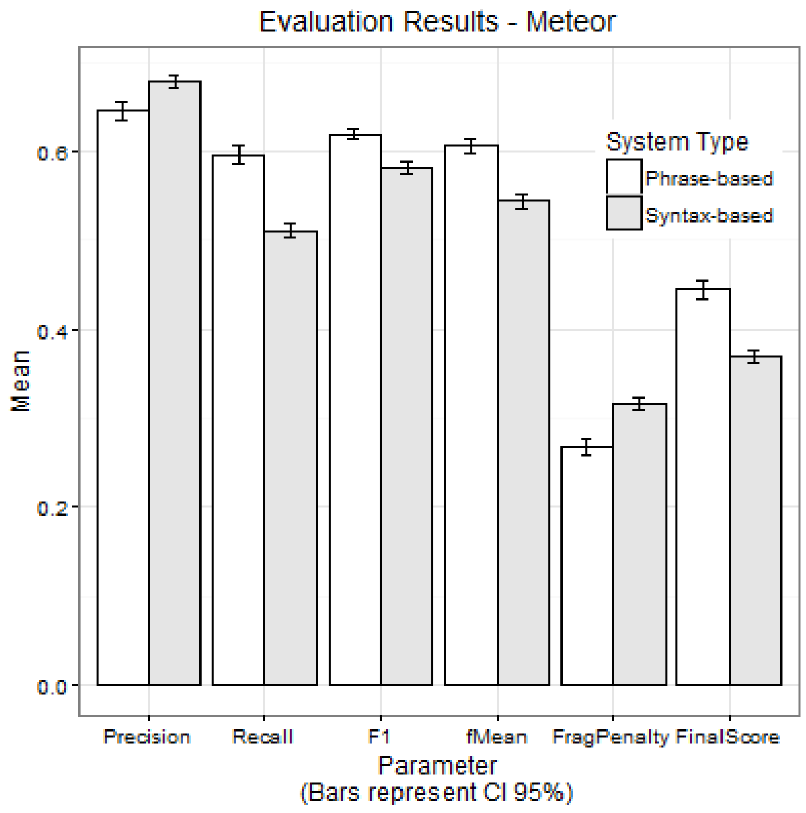


Figure 3.3: Results for the evaluation using Meteor.

---

quality of the syntactic classification performed by the parser chosen. Figure 3.4 shows three examples of sentences with classification problems. There are evident errors in the classification of verbs, names and articles. A recurring problem is the inadequate classification of the names of drugs.

```
(ROOT (S
   (NP (N Daniel))
   (VP (V deve) (VP (V' (V tomar) (ADV agora))
   (NP (NP (ART o) (N comprimido))
   (S (VP (V Duphaston) (NP (ART uma) (N' (N vez) (CP (NP (REL que))
   (S (VP (V são) (NP (CARD vinte) (N horas))))))))))))

(ROOT (S
   (NP (NP (N' (N No) (N fim) (N do) (N jantar)))
   (NP (N' (N dona) (N Inês))))
   (VP (V aplique) (NP (ART a) (N' (N pomada) (A Fucithalmic))))))

(ROOT (S
   (S (CONJ Quando) (S (NP (CL se)) (VP (V levantar)
      (NP (N' (N senhora)
      (N Pereira) (N tome))))))
   (S (NP (ART as) (N' (CARD duas) (N gotas)))
     |(VP (V' (V Neo-Sinedrina)
      (ADV depois))
      (NP (N' (N repita) (N ao) (N almoço)))))))
```

Figure 3.4: Examples of word classification errors made by the Stanford parser that, we believe, contribute to the inferior performance of the syntax-based systems.

The small extension of confidence intervals shows a small variation of the results with the division of the corpus used for testing.

From the combination of the results obtained by the two evaluation metrics it becomes clear that phrase-based systems performed better in the selected Data2Text task.

Results of evaluation by humans

The results of the evaluation by humans are summarized in Figures 3.5 to 3.7. In each figure, the counts for all possible answers are shown, comparing the results obtained by the two types of systems.

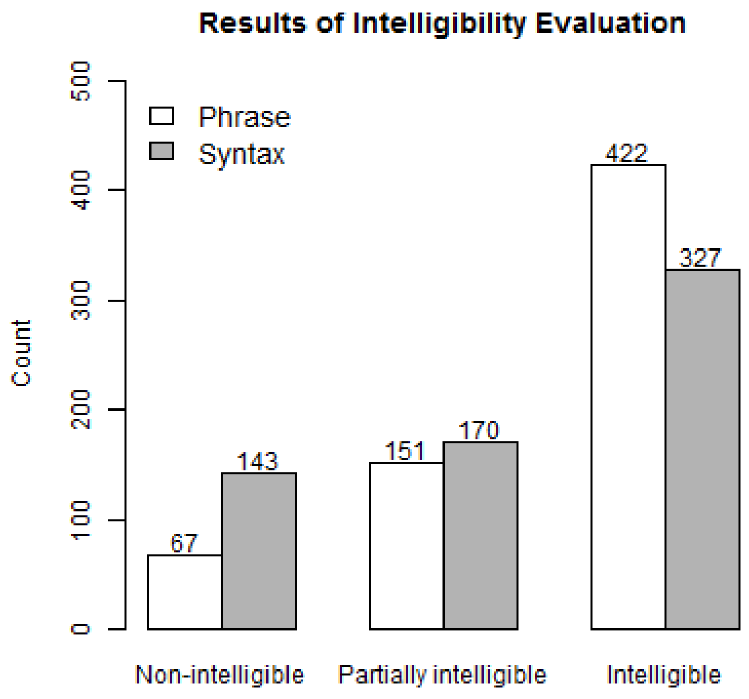


Figure 3.5: Distribution of the answers to the question on intelligibility of the sentences. The darker bars refer to the syntax-based system.

In terms of intelligibility, the phrase-based system obtained the higher number of responses indicating intelligible phrases with a difference of 95 responses, which means an average of 8.7 more positive responses per evaluator. The difference between the two systems of evaluations indicating partial intelligibility is low. The worse performance of the syntax-based system is found in the higher number of sentences evaluated as non-intelligible. While the phrase-based system has an average of 6 non-intelligible phrases per evaluator, the syntax-based system shows an average value more than twice that (13). Considering that 64 sentences were evaluated for each system, these numbers correspond to, respectively, 9% and 20% of non-intelligible sentences. On the positive side, an average of 60% and 46% of evaluated sentences were considered intelligible by evaluators.

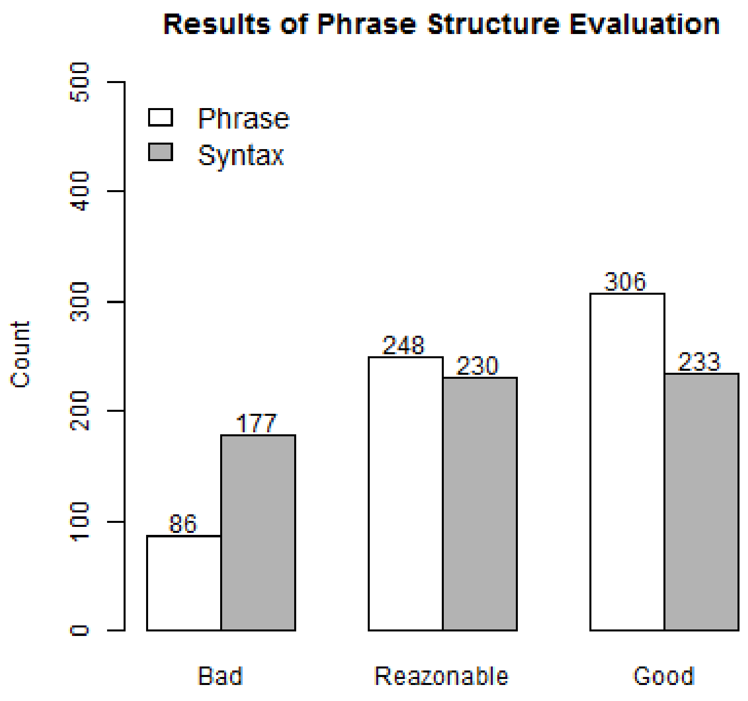


Figure 3.6: Results for sentences structure evaluation. The darker bars refer to the syntax-based system.

In terms of sentence structure (Figure 3.6), the best system continues to be the phrase-based one. Once again, the major differences occur in the extremes (bad structure and good structure).

In the overall quality assessment (Figure 3.7), the phrase-based system shows a higher number of evaluations obtaining the values generally considered a positive rating (3 or more) when compared to the syntax-based alternative. The system based on syntax obtained a much higher number of evaluations with the lowest value on the scale (1). Both systems feature a distribution of ratings throughout the 5 scale values, with a tendency for values between 3 and 5 in the case of the phrase-based system. It is important to note that on average, for the best system, 46% of the sentences were assessed as good or excellent and 25% as excellent.

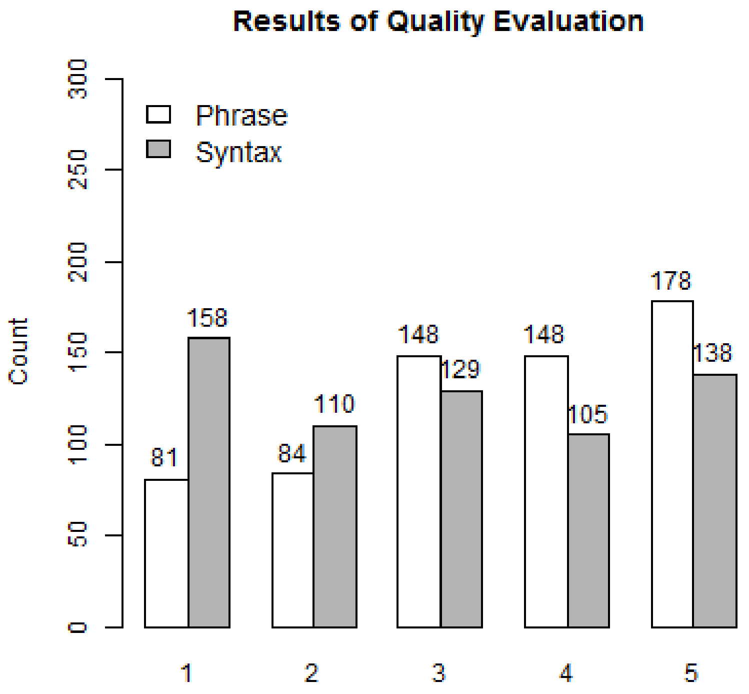


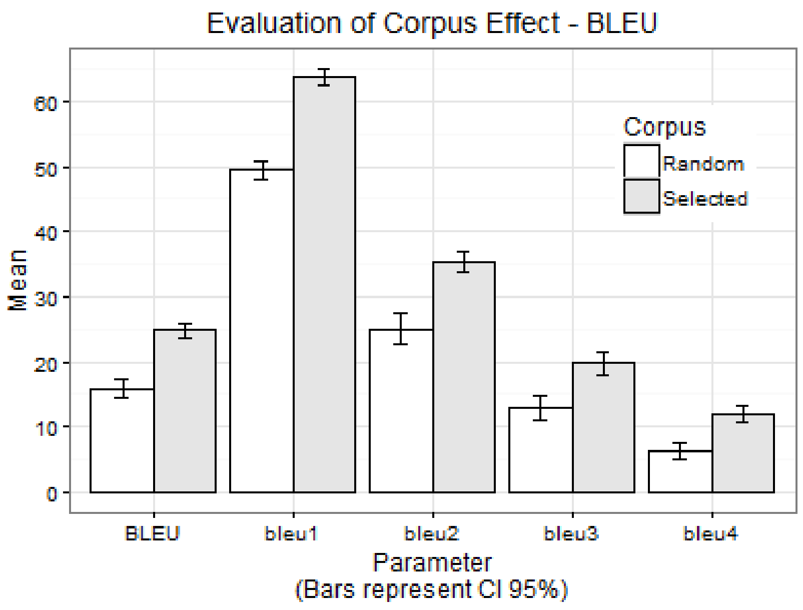
Figure 3.7: Results for overall quality evaluation. The darker bars refer to the syntax-based system.

3.3.3 Effect of the corpus division method

In order to rule out possible effects of the way the corpus was initially divided on the results (see Section 3.2.3), a new experiment was created, where the corpus was divided randomly into 10 new sets, with a similar number of phrases. For this experiment, taking into consideration the best results initially obtained, it was decided only the phrase-based systems would be used. New systems were trained and tested using the new randomly created sets, in the same way as the previous experiment.

Figure 3.8 shows the differences in the results obtained with the two methods of corpus division. It is worth noting that for both sets of metrics, assessed in 10 test sets, performance is better when using the split method proposed in this chapter. This difference is statistically significant for a confidence level of 5% (as 95% confidence intervals do not overlap).

Taking into account the superior results with non-random division of the corpus, the decision was made not to carry out any more training and testing with randomly divided corpora.



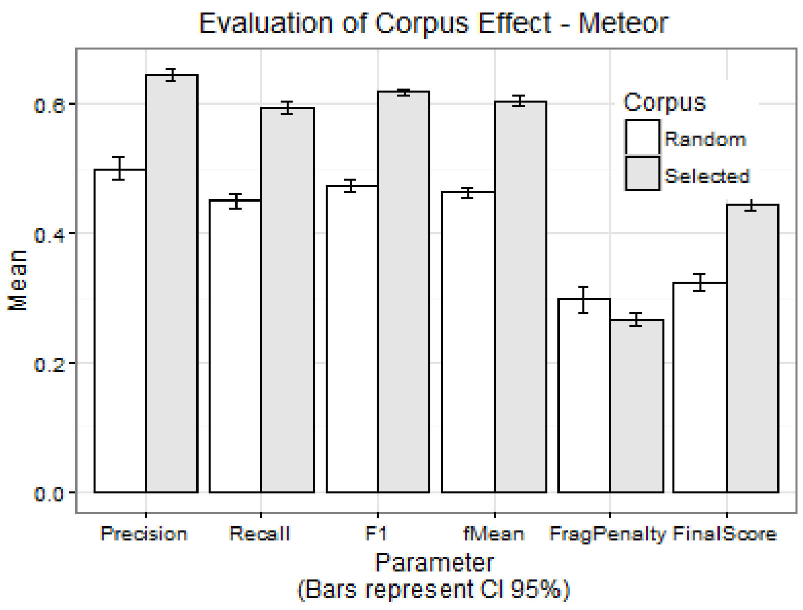


Figure 3.8: Results obtained with the two methods of corpus division used. The results only refer to phrase-based systems.

3.3.4 Variability of generated sentences

A good system based on templates should interact with human beings with great naturalness, variability and quality. However, for this to happen, it is necessary to make big investments in terms of time or resources.

The experiments carried out with our system showed that, for a similar input vector, correct and distinct answers are produced, which are not part of the initial corpus. This type of generation easily creates new answers, providing variability in interactions with the user. Table 3.7 presents two examples where, for similar input, different sentences are obtained. In examples 1 and 2, only the name of the person changes. In examples 3, 4 and 5, only the name of the medicine varies, and consequently, its form and method of intake.

Table 3.7: Example of output generated from similar inputs.

#	Example
1	pessoa18n saudacao_m pessoa0a medicamento24 tipo3 tomar1 cor00 dose0 freqtoma20 Senhor Daniel aplique a pomada Fucithalmic que são vinte horas
2	pessoa18n saudacao_m pessoa5a medicamento24 tipo3 tomar1 cor00 dose0 freqtoma20 Senhor Daniel Costa deve aplicar a pomada Fucithalmic que são vinte horas
3	pessoa18n saudacao_m pessoa0a medicamento24 tipo3 tomar1 cor00 dose0 freqtoma20 Senhor Daniel aplique a pomada Fucithalmic que são vinte horas
4	pessoa18n saudacao_m pessoa0a medicamento3 tipo1 tomar2 cor00 dose0 freqtoma20 Senhor Daniel não se esqueça de tomar o comprimido Ibuprofeno são vinte horas
5	pessoa18n saudacao_m pessoa0a medicamento12 tipo1 tomar2 cor00 dose0 freqtoma20 Senhor Daniel tome o comprimido de Nicotibine são vinte horas

3.3.5 First integration of the Data-to-Text module in applications

An early version of the phrase-based system presented here has been integrated in a real application for the first time, an assistance tool aimed at seniors called Medication Assistant. This system for Smartphones, developed under the Smartphones for Seniors project, has been described and evaluated in (Teixeira et al., 2013b, Ferreira et al., 2013, Ferreira et al., 2014).

3.4 Discussion

The results obtained by the best system, in terms of intelligibility and quality of the generated phrases, indicate that the chosen approach and the systems developed can generate sentences of similar quality to those produced by humans. However, when they fail, the generated phrases can be completely unintelligible.

The results of the evaluation, including automatic evaluation and evaluation by humans, showed that the phrase-based system has the best performance. This type of system is able to generate a good percentage of intelligible or minimally intelligible sentences (less than 10% of non-intelligible phrases), with a good percentage of generated sentences receiving overall quality evaluations of ‘good’ or higher. The biggest limitation of the developed systems is their unpredictability in terms of the quality of results.

Our results for phrase-based systems are in line with those reported by Langner for the MOUNTAIN system (Langner, 2010), presented in Tables 3.8 and 3.9. No comparison can be made for the syntax-based systems as MOUNTAIN only used the phrase-based.

Table 3.8: Published evaluation results for MOUNTAIN system – BLEU (from (Langner, 2010, p.73).

System	1-gram	2-gram	3-gram	4-gram	5-gram
Baseline	0,3198	0,1022	0,0525	0,0300	0,0202
Rating > 1	0,4376	0,1729	0,1079	0,0746	0,0597
Rating > 2	**0,4491**	**0,1919**	0,1169	0,0872	0,0747
Rating > 3	**0,4742**	**0,1963**	0,1212	0,0866	0,0722
Rating > 4	**0,4596**	0,1762	0,1023	0,0693	0,0611

Table 3.9: Published evaluation results for MOUNTAIN system – Meteor (from (Langner, 2010, p.75)).

System	Precision	Recall	f1	Total
Baseline	0,4225	0,2013	0,2727	0,1950
Rating > 1	0,4489	0,2097	0,2859	0,2028
Rating > 2	0,4533	**0,2248**	**0,3009**	**0,2218**
Rating > 3	**0,4834**	0,2148	0,2974	0,2146
Rating > 4	0,4481	0,2030	0,2794	0,1971

Comparing our results from Figure 3.2 with Table 3.8 it can be observed that our system produces slightly higher BLEU scores than MOUNTAIN (any version) when comparing Bleu1, Bleu2, Bleu3 and Bleu4. Table 3.9 shows the results of MOUNTAIN's assessment by Meteor. Comparing our results in Figure 3.3 limitations of our systems, their performance is satisfactory and comparable to the performance of similar systems.

One of our goals is to generate a system which requires only a low amount of effort to obtain corpora. With this in mind, our systems were able to generate an interesting set of sentences, even though they were trained with only a very small corpus, derived from little more than 100 phrases actually written by humans. This fact demonstrates not only the potential for improvement, by increasing the corpus, but also the potential to build minimally useful systems with very little investment in the creation of corpora.

The performance of the simplest system, based on phrases, should not be interpreted as a definite indication of the inadequacy of the syntactic-based systems in general for data-to-text tasks, as some factors affected their performance. Potential reasons for this inferior performance include: (1) the reduced size of the corpus, which may be insufficient to properly train the syntactic-based models, which are certainly more demanding in terms of the number of examples; (2) the negative effect of syntactic analysis errors. As for this second problem, we are convinced that with the use of a parser which does a better job of classifying words with regards to their syntactic function, the performance of the syntax-based system will improve.

On the other hand, the fact that the two systems often generate very different phrases can be used as an advantage in the creation of a system in which the results of both are the subject of an evaluation process and the best phrase is selected.

To divide the corpus in order to train the 10 test systems, an alternative process to the usual random division was developed, which contributed to a better performance of the systems being tested. As an expansion process was used in the creation of the corpus, the usual method of random division is not the best solution, as it does not ensure that examples derived from the same sentence stay properly divided between training and test sets. The method proposed avoids a low presence of examples resulting from a base sentence in the training set.

The evaluation, as expected, proved to be a very complicated task, with the automatic metrics showing great difficulties in providing adequate information. Only by using two sets of metrics combined with evaluation by humans was it possible to arrive at a minimally clear vision of the systems' performance.

3.5 Conclusion

In this chapter a first proof-of-concept Data-to-Text system for a specific domain was presented, developed by adopting translation-based methods. Variants of the system were created based on two different translation methods: phrase-based and syntax-based (also designated as tree-based). Systems were created resorting to a small corpus defined, collected and processed for this purpose.

The systems developed were evaluated using commonly used metrics, BLEU and Meteor, as well as human evaluations of intelligibility, sentence structure and overall quality. Results showed that for this domain and the small amount of data available for training, phrase-based systems show a better performance, and sentences generated can have the variability required to be used in human-computer interaction.

Results also showed that the unpredictable quality of the generated output is a major limitation, with a fair percentage of sentences not reaching the required level of intelligibility, structure and overall perceived quality. In order to tackle this limitation, new systems combining the best translation-based systems with other methods of natural language generation – template-based methods – were explored next, which is the subject of the next chapter.

Chapter 4

Hybrid Data2Text system: A second system combining translation-based with templates

Despite the fact that the system described in the previous chapter provides sentences with the variability which is essential to improve interaction, this does not come without a cost. In contrast with template-based systems, this system produces a highly heterogeneous output. This heterogeneity is a major problem, since it could possibly provide low-quality sentences to users. By this we mean unintelligible sentences or ones which convey incorrect information.

In this chapter we propose combining a translation-based NLG system with a classifier module capable of providing information on the Intelligibility or Quality of the sentences. Sentences marked as unacceptable will be replaced by ones generated by a template-based system.

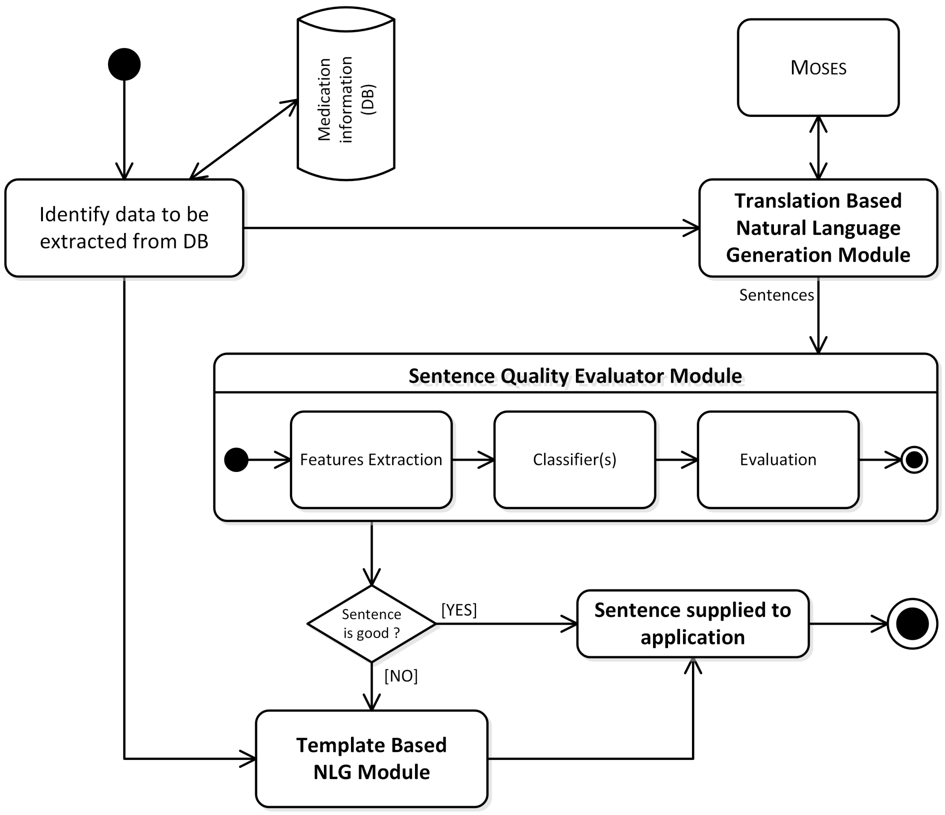
The classifier module and the templates module are the main focus of the chapter. The evaluation and classification of produced sentences were done with free tools which are available to the public. It combines the extraction of linguistic features with classifiers trained in a manually annotated corpus. The template-modules were developed with XML and XSLT. One reason was the issue of availability as well as the zero-cost technology. Another, was the ease of development.

4.1 Architecture of Hybrid Data2Text system

The need to evaluate sentences produced, and in some cases, replace them with new sentences, was the motive behind an improvement to the system's architecture, with the addition of new modules. The first module added was the Sentence Quality Evaluator module. It is responsible for the evaluation of sentences. It takes, as input, the sentence produced by the Data2Sentences module (now, renamed as the Translation Based Natural Language Generation Module), and analyzes its syntax and grammar, and determines whether it is acceptable.

If a sentence is marked as unacceptable, it must be replaced by a new one. The mission of the Template-based module is to provide a substitute sentence, even if the variability of produced sentences is affected. It is very important that the new sentence expresses the same data as the initial one. Therefore there must be a connection to the component that decided which data should be extracted from the database.

This new system version is called 'Hybrid Data2Text system', due to the multiple approaches to producing a sentence. Figure 29 shows its architecture. The new modules are explained throughout the next sections.



Medication information (DB)
MOSES
Identify data to be extracted from DB
Translation Based Natural Language Generation Module
Sentences
Sentence Quality Evaluator Module
Features Extraction
Classifier(s)
Evaluation
Sentence is good ?
[YES]
[NO]
Sentence supplied to application
Template Based NLG Module

Figure 4.1: Architecture of the Hybrid Data2Text system

4.2 Sentence Quality Evaluator module

Human languages are so rich and complex, that evaluating a simple sentence, no matter what language it was produced in, is a very difficult task. In this module, we split the evaluation into three sub-tasks. First, Feature Extraction. It is responsible for counting the features used later by classifiers to evaluate sentences. Second, Classifiers. They are responsible for classifying each sentence. The third part is presented at section 4.4 (Evaluation of the quality of generated output), which presents the results obtained by using classifiers and features extraction.

4.2.1 Feature Extraction

After Moses translation, every utterance produced must be evaluated, but the evaluation of human texts is a demanding task. If humans can generally judge the overall quality of a sentence by simply reading it, to machines this is impossible. For machines, one possible option is to count relevant words in sentences, and compare their values to reference texts to have a clue regarding sentence quality. We emphasize the word 'clue'. Automatic evaluations are tools which aid the decision-making process, but they are not 100% accurate. They need to be flexible and versatile in determining the correctness of a sentence, taking into account that human sentences are a very rich and distinctive 'product'.

In his Master's thesis (Felice, 2012) defined a set of 147 different features, related to translation to and from the Spanish language. These features identify and count the relevant words in input and output languages, present in the translation process. These sets are classified and organized according to their use: linguistic features – features that rely on some linguistic knowledge of the language they refer to; shallow features – also known as 'non-linguistic' features, refers to features that are not related to linguistic knowledge. They could also identify statistical data such as n-gram probabilities.

Our evaluation begins with the feature extraction for each of the sentences produced. For our work, based on Felice (2012), a set of 11 features was selected. The selection was determined by the structure of our corpus, where only the output language is a human language. Another restriction was related to the availability of tools for processing the Portuguese language.

The selected features were: `target-cont` – counts the number of content words on a sentence, i.e., the number of relevant words, such as nouns, full verbs and adjectives;

`target-cont-words-pcent` – expresses the percentage of content words to the total of words in a sentence; `target-func-words` – counts the number of function words, i.e., determiners, pronouns, prepositions, conjunctions, auxiliary verbs and adverbs of degree; `target-func-words-pcent` – expresses the percentage of function words to total of words in a sentence; `target-n-pcent` – expresses the proportion, in percentage, of nouns including both common and proper nouns; `target-v-pcent` – like previous features, expresses the proportion, in percentage, of verbs including both full and auxiliary verbs; `target-np` – counts the total number of Noun Phrases (NP) of a sentence. Unlike Felice (2012), we count all NPs in a sentence; `target-vp` – counts the total number of Verb Phrases (VP). Like in the previous feature, we count all VPs; `target-pp` – counts the total number of Prepositional Phrases (PP). As in previous features, we count all PPs. `target-ptree-width` and `target-ptree-depth` analyzes the structure of a sentence. 'Tree width' computes the number of root node children and 'tree depth' computes the longest path from the root node to the leaves. Figure 4.2 presents a sample of output from *Translation Based NLG Module* and the corresponding tree. As this system is an evolution of MEDICATION2PT, the samples produced were related to its domain. Is possible to see that `target-ptree-width` as the value `2`, as it measures the number of first subdivisions of sentence, and `target-ptree-depth` as the value `5`, as it measures the deepest token from the beginning of sentence.

```
(ROOT
  (S (NP (N' (N Maria)
             (N ao)
             (N pequeno-almoço)))
     (VP (V aplique)
         (NP (ART a)
             (N' (N bomba)
                 (N Nasomet)))
     )
   )
 )
```

target-ptree-width = 2
target-ptree-depth = 5

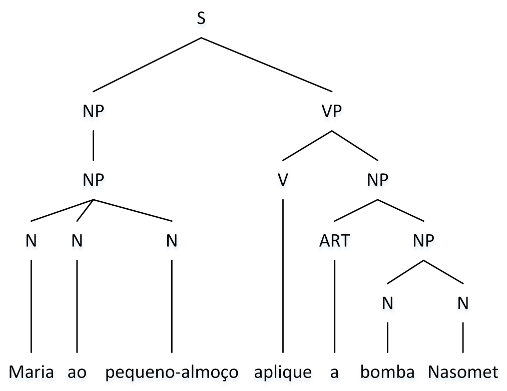


Figure 4.2: A sample of an output from *Translation Based NLG Module* and the corresponding tree
(in English: *Mary, at breakfast, apply the Nasomet pump*)

The parsing of sentences is, first of all, made with a POS[1] tagger, which classifies words and identifies partial phrases (NPs, VPs or PPs) in each sentence. Experiences were made with TreeTagger (Schmid, 1995, Schmid, 2015) and Stanford Parser (University of Stanford, 2015). Each of them was adapted to process Portuguese. Stanford

[1]POS – Part-Of-Speech

Parser was adapted by the LX-CENTER (Branco and Silva, 2004), at Lisbon University. TreeTagger was adapted by Mário Rodrigues (Rodrigues, 2013) in the course of his PhD work.

To process the features extraction, a JAVA program was made. After classification, the set of features is computed, counting each feature from every sentence. Stanford Parser proved itself a better parser for our needs than TreeTagger, as it is the only one in which it is possible to obtain the full set of features.

4.2.2 Classifiers

Taking in consideration the small dataset – we only have about 640 sentences –, as classifiers, besides the commonly used `SVM`, two others were selected: the lazy `IB1` and the tree based `Random Forest` (Hall et al., 2011). Support Vector Machines (SVMs) (Witten and Frank, 2005) are based on the concept of decision planes that separate between a set of objects which have different class memberships. The original objects are mapped using a set of mathematical functions (known as kernels) to make the mapped objects linearly separable. This way, instead of finding a complex separation curve, it is only necessary to find an optimal line that separates the classes. We used libSVM implementation, running under Weka (Hall et al., 2009), with the default configuration.

IB1 is the simplest of Instance-Based Learning (IBL) algorithms. IBL algorithms are based on the Nearest-Neighbor method. Unlike the Nearest-Neighbor method, which is not incremental and tends to maintain consistency with the training set, and is prone to problems related to noise provided by samples, IBL algorithms are incremental and their main goal is the maximization of prediction, based on successive iterations (Aha et al., 1991). Prediction is based on determining which training set member is closest to an unknown test instance. Once it is determined, its class is assigned to the test instance. Usually, as a distance function for prediction, the normalized Euclidean distance is used. Under special circumstances, the Manhattan metric or other metrics can be used (Hall et al., 2011, pg.131).

Random Forest (Breiman, 2001) is a statistical tool used to predict values. After the collection of sample vectors, they are divided into multiple decision trees. Each tree is obtained randomly and independently, and takes, at the most, 2/3 of the sample's dimension. Hence the name of this method. When a prediction is needed, each decision tree is asked to predict a value. After that, the most predicted value is the chosen one. This algorithm assumes that (1) most decision trees are able to perform a correct prediction, based on sample vectors; (2) errors provided by decision trees, do not occur in the same position.

4.3 Template-Based module

4.3.1 Overview

When MOSES cannot produce an acceptable utterance, the *Templates module* is the last chance of producing one. In fact, this module is key to the hybrid approach of our system. It uses the same data as the Data2Sentence module, but in a different manner. In it, the data will be processed by a stack of templates to produce the final result.

4.3.2 Master Requirements

The main requirement of this module is that it must be context-free. The aim is to produce a module that can be used in diverse scenarios, by simply adapting the templates. To achieve that, a set of free and easy to use tools to classify words and sentences are used. We have chosen the FreeLing (Padró, 2011, Padró and Stanilovsky, 2012) and jSpell + Lingua (Simões, 2016). All data is expressed in XML files. XML was chosen because it is simple and easily read, by both humans and computer programs. Since data is expressed in XML, using XSLT as *templates* was a natural choice. It allow us to produce several XSL files, each one aimed at solving a specific problem. The use of these files, in a specific order, produces the desired utterance.

4.3.3 Templates Module Architecture

Figure 4.3 shows the architecture of the *Templates module.* When one decides to use this module, the first action is to process the *data extraction.* Data extraction consists of querying the database, to find values to use on templates. The database could be a SQL relational database, or any other kind of data source.

After extraction, this data is used to produce an XML file, which is related to one utterance. That file is the first document on the *templates pipeline.* In Figure 4.4 a full sample of it can be seen. As expressed before, the sample presented, and those that follow, are related to the MEDICATION2PT project. However, we reinforce that nothing in our approach is tied to that project. This file has two main sections:

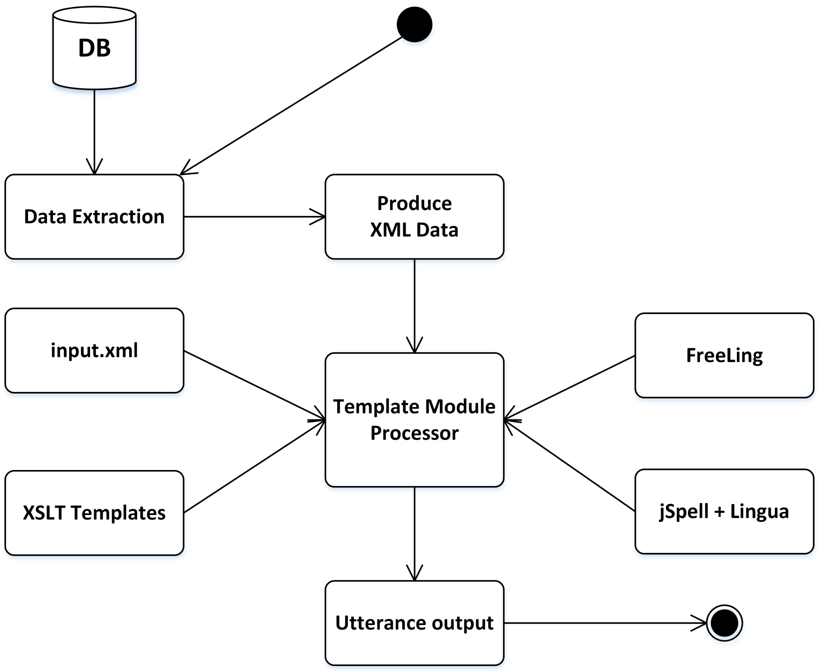


Figure 4.3: General *Templates module* architecture

<`withoutProcessing`> and <`processing`>. In the first (see Figure 4.5), the data that is not intended to be changed by the *Template Module Processor* is described. This data is to be used as it is. Each tag is identified by a `name` and has a `value`. If necessary, this section may be empty, without any values.

```
<?xml version="1.0" encoding="utf-8" ?>
<data>
   <withoutProcessing>
      <tag name="saudacao" value="Sr."/>
      <tag name="nome" value="Fernando"/>
      <tag name="apelido" value="Silveira"/>
      <tag name="medicamento" value="Fluconazol"/>
   </withoutProcessing>
   <processing>
      <tag name="freqToma" value="almoço" dependency=""/>
      <tag name="formaToma" value="tomar" dependency=""/>
      <tag name="dose" value="2" dependency="tipo"/>
      <tag name="tipo" value="cápsulas" dependency=""/>
      <tag name="cor" value="brancas" dependency="tipo"/>
   </processing>
</data>
```

Figure 4.4: XML file with data extracted from database, aimed to create the sentence

The <`processing`> section (see Figure 4.6) describes the data which is meant to be processed. Here, we are going to classify each term by number (singular/plural), gender (feminine/masculine/common), lemma and type (adjective, noun, verb, etc.).

Each tag is identified by three types of data: `name` and `value`, as presented in 'without processing' section, they describe the name of data and its value. The third parameter – `dependency` – expresses the relation, if any, between that tag and another tag, in this section. That dependency can be in terms of number, gender or to the noun with the related tag.

```
<withoutProcessing>
   <tag name="saudacao" value="Sr."/>
   <tag name="nome" value="Fernando"/>
   <tag name="apelido" value="Silveira"/>
   <tag name="medicamento" value="Fluconazol"/>
</withoutProcessing>
```

Figure 4.5: *Without processing* section.
Sample extracted from MEDICATION2PT project.

```
<processing>
   <tag name="freqToma" value="almoço" dependency=""/>
   <tag name="formaToma" value="tomar" dependency=""/>
   <tag name="dose" value="2" dependency="tipo"/>
   <tag name="tipo" value="cápsulas" dependency=""/>
   <tag name="cor" value="brancas" dependency="tipo"/>
</processing>
```

Figure 4.6: *Processing* section. Sample extracted from MEDICATION2PT project.

In the <`withoutProcessing`> section we store data that, due to its nature, does not need to be processed, because we know all we need to know about it. That way, we save processing time, and avoid potential errors that may occur. In the <`processing`> section we store all data that we need to learn more about.

All the work described here is done by the *Template Module Processor.* After the XML data file is prepared, it is possible to process it. The Template Module Processor requires several files to be configured. One of the most important is the **input.xml** file. This file (see an example in Figure 4.7) describes all files that are used by the templates module, as well as their location. The <`data`> section describes the name of the XML data file and its location. It also identifies where the templates files and 'result file', with the utterance produced, are saved. The other section – <`processing`> – identifies all template files used by the module. The order presented is the order used. As previously described for the other files, the input.xml tags have two distinct terms: `key` and `value`, where the name of the data and its value are described.

One of the main parts of this module is the set of templates, created with XSLT

```
<?xml version="1.0" encoding="utf-8" ?>
<templates>
   <data>
      <!-- XML data file name -->
      <tag key="dataFile" value="dadosXML_v1.xml"/>
      <!-- XML data file location -->
      <tag key="dataFileLocation" value="Templates-dados"/>
      <!-- XSL files location -->
      <tag key="templatesLocation" value="Templates-xsl2dados"/>
      <!-- 'result' file location -->
      <tag key="resultsLocation" value="Templates-resultados"/>
   </data>
   <processing>
      <!-- XSL files used on Templates Module.
           Files are used by this order -->
      <tag key="process" value="frase.xslt"/>
      <tag key="process" value="fraseFinal.xslt"/>
      <tag key="process" value="resultado.xslt"/>
   </processing>
</templates>
```

Figure 4.7: *input.xml* file sample

language[2]. XSLT is the natural choice to process and change XML data. Each XSL file is used to solve a particular problem related to generating the utterance. The use of XSLT to produce templates is well documented in literature, as can be seen for instance in (Wilcock, 2001, Grover et al., 2002, Stenzhorn, 2002, Foster and White, 2004, Wilcock, 2005, Salem et al., 2014, Chiarcos et al., 2015).

The *Template Module Processor* is responsible for the coordination of the entire process. Its work-flow is represented in Figure 4.8.

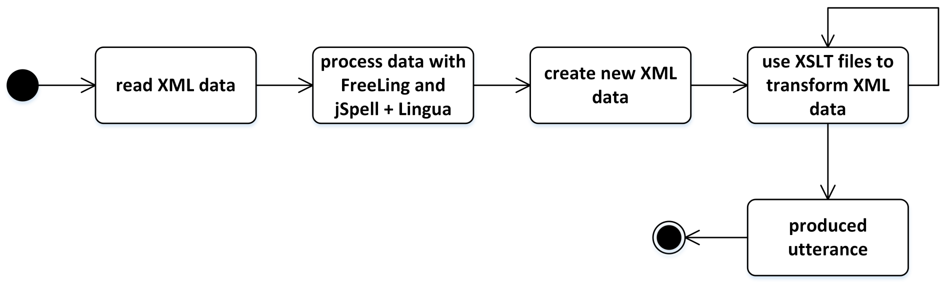

read XML data
process data with FreeLing and jSpell + Lingua
create new XML data
use XSLT files to transform XML data
produced utterance

Figure 4.8: The *Templates Module Processor* work-flow

The first job is a very simple one: to read the XML data file, whose name and location is expressed in the *input.xml* file.

The next job is one of the main tasks of this module – to produce new XML data. The data in the <withoutProcessing> section remains untouched. Each value of each

[2]https://www.w3.org/TR/xslt/

tag in the <processing> section is classified with the FreeLing tool (Padró, 2011, Padró and Stanilovsky, 2012), which has a Portuguese language classifier. The classification is expressed by determining the POS tagging, lemma, class, gender and number `values`.

This classification is used to help the XSLT templates do their job. The new XML data is obtained from data from the <withoutProcessing> section as well as the processed data. Figure 4.9 presents the result of processing the data in Figure 4.4. Six new parameters are added to each tag: `class` – represents the class of the value (noun, verb, adjective, etc.); `type` – represents the subclassification of the class; `lemma` – the lemma of the value; `alternativeValue` – alternative value to be used by XSLT template, if necessary, to adjust gender and number; `gender` – feminine, masculine or common; and `number` – singular or plural.

The `terms` parameter, at the <processing> tag (see Figure 4.9), represents the values to be used by the XSLT templates. It is composed of a succession of 0 (zero) or/and 1 (one). 'Zero' means that there is no value to process, and 'one' means the opposite. In this example, the <processing> section has five tags. Hence, the `terms` parameter shall consist of five digits. The order of each digit is the order of corresponding tag. If the value of one tag is marked as being dependent on another tag, the correspondence in gender and number between them is analyzed. If necessary, the jSpell + Lingua tool (Simões, 2016) is used to determine the list of derived words of the dependent value. These derived words are then processed by FreeLing to help determine the best choice.

The final task is to use the sequence of XSL files to transform the new XML data, and produce the final utterance. We do not expect to have one simple XSL file. The idea is to use several files, each one to solve a particular task. In this case, we used three sets of XSL files. Each one uses as input, the output of previous set. The first one is used to pre-process the tags used to produce the final utterance. The second is responsible for producing the utterance. Figure 4.10 and Figure 4.11 show the XML data, after the first and second steps using the XSLT templates.

The last XSLT template is responsible for extracting the utterance words, forming the desired *utterance*. This utterance can be written by the output device or simply used elsewhere. Table 4.1 presents the data used and the final result obtained by the Templates Module.

```
<?xml version="1.0" encoding="utf-8" ?>
<data>
   <withoutProcessing>
      <tag key="saudacao" value="Sr."/>
      <tag key="nome" value="Fernando"/>
      <tag key="apelido" value="Silveira"/>
      <tag key="medicamento" value="Fluconazol"/>
   </withoutProcessing>
   <processing terms="11111">
      <tag key="freqToma" value="almoço" class="nome" type="comum"
           lema="almoço" alternativeValue="almoço"
           gender="masculino" number="singular"/>
      <tag key="formaToma" value="tomar" class="verbo" type="principal"
           lema="tomar" alternativeValue="tomar" gender="" number=""/>
      <tag key="dose" value="2" class="numeral" type=""
           lema="2" alternativeValue="2"
           gender="feminino" number="plural"/>
      <tag key="tipo" value="cápsulas" class="nome" type="comum"
           lema="cápsula" alternativeValue="cápsula"
           gender="feminino" number="plural"/>
      <tag key="cor" value="brancas" class="adjetivo" type="qualificativo"
           lema="branco" alternativeValue="branca"
           gender="feminino" number="plural"/>
   </processing>
</data>
```

Figure 4.9: The new XML data, after being processed by FreeLing and jSpell+Lingua

4.4 Evaluation of the quality of generated output

Over the next section we present the evaluation of sentences produced by the first module of *Translation Based NLG Modules*. It might be assumed that a full evaluation of our complete system would be carried out – the sentences produced by Data2Sentences module, based on Moses, and the alternative sentences produced by the Templates module, based on XML and XLST. Certainly that would have been ideal, but to do so would have required a very large amount of time.

'Templates' are a well-studied field of Natural Language Generation. We only want to use them, and do not seek to prove anything further about them. It is our belief, based on the way we built our template module, that the sentences they produce are understandable and well formed.

Our main emphasis is on automatic translation. We re-affirm that the aim of this evaluation is to assess the quality of sentences produced by the Data2Sentences module. Only if a sentence is marked as 'to be replaced', is the Templates module used. This means the Sentence Quality Evaluator module is a key part of our system, since it enables this 'hybrid' behavior.

```
<?xml version="1.0" encoding="UTF-8"?>
<sentence>
   <salutation>
      <tree label="N"> Sr.</tree>
      <tree label="N"> Fernando</tree>
      <tree label="N"> Silveira</tree>
   </salutation>
   <medicine terms="111111">
      <tree label="N" tag="efetuarToma"> almoço</tree>
      <tree label="V" tag="formaToma"> tomar</tree>
      <tree label="CARD" tag="dose" genero="feminino"
               numero="plural"> 2</tree>
      <tree label="N" tag="tipo"> cápsulas</tree>
      <tree label="A" tag="cor"> brancas</tree>
      <tree label="N" tag="nomeMedicamento"> Fluconazol</tree>
   </medicine>
</sentence>
```

Figure 4.10: The XML data, after being processed by the first set of XSLT templates

Table 4.1: Data used, and final result, from the Templates Module.

data:	Name:	Fernando
	Last name:	Silveira
	Courtesy:	Sr.
	Medicine:	Fluconazol
	Time to take medicine:	almoço
	Way of taking medicine:	tomar
	Quantity of medicine:	2
	Medicine type:	cápsulas
	Color of medicine:	brancas
result:	Sr. Fernando Silveira deve tomar as 2 cápsulas brancas de Fluconazol ao almoço	

This section concludes the Sentence Quality Evaluator module. The *features* and *classifiers* mentioned in the section 4.2 are used here.

4.4.1 Corpus/database

For the evaluation of the output sentences, a corpus of 640 sentences produced by the Data2Sentences module were collected and manually annotated regarding Intelligibility, Grammaticality and Overall Quality. For Intelligibility and Grammaticality, the annotation was expressed on a 3-point Likert scale: not good, partially good or good. Overall Quality was annotated with a 5-point Likert scale, from bad to very good. Table 4.2 shows the Likert scales used for these annotations.

```
<?xml version="1.0" encoding="UTF-8"?>
<tree label="TOP">
   <tree label="S">
      <tree label="NP">
         <tree label="N1">
            <tree label="N"> Sr.</tree>
            <tree label="N"> Fernando</tree>
            <tree label="N"> Silveira</tree>
         </tree>
      </tree>
      <tree label="VP">
         <tree label="VP">
            <tree label="V"> deve</tree>
            <tree label="V"> tomar</tree>
         </tree>
         <tree label="NP">
            <tree label="ART"> as</tree>
            <tree label="CARD"> 2</tree>
            <tree label="N1">
               <tree label="N1">
                  <tree label="N"> cápsulas</tree>
                  <tree label="A"> brancas</tree>
               </tree>
               <tree label="PP">
                  <tree label="P"> de</tree>
                  <tree label="N"> Fluconazol</tree>
               </tree>
            </tree>
         </tree>
         <tree label="VP">
            <tree label="N"> ao</tree>
            <tree label="N"> almoço</tree>
         </tree>
      </tree>
   </tree>
</tree>
```

Figure 4.11: The last XML data, after being processed by the second set of XSLT templates

The aim of previous evaluations was to determine the quality of produced sentences. The next step, is to determine the acceptability of sentences. This led to a reclassification of the obtained classifications into a two-score class. Every evaluation was converted to 0 (zero) or 1 (one) values. `Zero` means that the sentence must be rejected, and `one` means that the sentence should be maintained. All sentences were automatically processed to create derived classifications considering only these 2 classes. Intelligibility and Grammaticality answers, annotated initially as 0 or 1, were assigned to the 0-class. If they had been annotated with the score 2, they were assigned to the 1-class. Overall Quality was converted too. Scores related to sentences annotated as

Table 4.2: Information on the manual classifications used in the corpus evaluation

Annotation	**Question**	**Possible answers**	**Description**
Intelligibility	Understandable?	0, 1, 2	0=No; 2=Yes; 1=Partially
Grammaticality	Structure of sentence?	0, 1, 2	0=Bad; 2=Good; 1=Reasonable
Overall Quality	Quality in general?	1, 2, 3, 4, 5	1=Bad; ... 5=Very Good (natural)

good or very good (4 or 5) were assigned to 1-class. The remaining sentences (1 to 3) constitute the 'no-good' sentences, and were assigned to 0-class. A final classification was also created, by performing a logical AND with the 3 derived classifications. This last classification helps identify sentences that are considered good in all aspects being evaluated. Table 4.3 shows a small sample of sentences and their classifications. Names ending in 2, like *Intelligibility 2*, express the two-score classes.

Table 4.3: Sentences and their corresponding annotations

Intelligibility	**Grammaticality**	**Overall Quality**	**Intelligibility 2**	**Grammaticality 2**	**Overall Quality 2**	**AND 2 of three**
Sentence: Não se esqueça senhor Mário que deve tomar três comprimidos de Maltofer						
2	2	5	1	1	1	1
Sentence: Dona Rosa a pomada branca outra almoço						
1	0	1	0	0	0	0
Sentence: Gustavo bombas de Seretaide ao jantar						
1	0	3	0	0	0	0
Sentence: Dona Rosa para ficar boa todos as manhãs ao levantar						
0	0	1	0	0	0	0
Sentence: César não esqueça tome quatro gotas de Neo-Sinedrina ao levantar						
2	1	5	1	0	1	0
Sentence: Senhor Rodrigo Alves deve tomar o rebuçado Tatum Verde de jantar						
2	1	4	1	0	1	0

In the first sentence of Table 4.3, it is possible to see that in all previous classifications it obtained the highest possible scores. Therefore all new classes have been scored as 1. The second sentence was considered a 'bad' sentence. It was considered 'partially' intelligible (Intelligibility=1), with 'bad' grammaticality and with the lowest overall quality. This implies that all of its corresponding new scores are 0. The third and fourth sentences have similar classifications to the second sentence. The fifth and sixth sentences, despite scoring high in intelligibility and overall quality, because the human evaluator scored grammaticality as 'reasonable', this forced the Grammatic2 to score 0. Consequently, AND2 classification is also 0.

Finally, all sentences of this corpus were processed in order to extract the features presented by subsection 4.2.1 (Feature Extraction). After that, seven different datasets were created by appending to the extracted features, one of each of the above classifications. For it to be possible to use them in Weka, all datasets were converted to the ARFF[3] format.

4.4.2 Datasets Balancing

As the obtained datasets had a clearly unbalanced number of samples for the possible classes, the SMOTE technique (Chawla et al., 2002) ("Synthetic Minority Oversampling TEchnique" – one of the techniques available in Weka) was adopted to balance the samples. After this, the order of the samples was also randomized.

4.4.3 Metrics

To automatically evaluate sentences, 3 ratios were computed:

- recall ratio (R):

$$R = \frac{tp}{tp + fn}$$

- precision ratio (P):

$$P = \frac{tp}{tp + fp}$$

- and, F-ratio (F):

$$F = \frac{2RP}{R + P}$$

with `tp` being the number of true positives, `fp` the number of false positives and `fn` the number of false negatives.

4.4.4 Results

The evaluation of these sentences, and consequently the work of this module, is based on human evaluations, features extraction and respective classifiers. These influencing factors were investigated using a 10-fold cross validation process, as we did before. From the 7 possible different types of system we could create, based on the 7

[3]An ARFF (Attribute-Relation File Format) file is an ASCII text file that describes a list of instances sharing a set of attributes (Weka, 2017).

human classifications available, we only took into account the ones related to Intelligibility, Quality and the logical AND. For Quality, due to the size of the dataset, we only considered the 2-class case.

The 4 selected datasets – Intelligibility (Int), Intelligibility 2 (Int2), Overall Quality 2 (Qual2), and the AND 2 (And2) – were processed by each one of the 3 selected classifiers (SVM, IB1 and Random Forest (RF)). The results obtained are shown in Table 4.4. Results are related to `true positives`, `false positives` and P, R and F values obtained by the formulas above. The ROC area (Fawcett, 2006) was also calculated.

Table 4.4: Evaluation results

System	**Number of Classes**	**Classifier**	**tp**	**fp**	**P**	**R**	**F**	**ROC Area**
Int	3	SVM	0,869	**0,068**	0,869	0,869	0,869	0,900
Int	3	IB1	0,840	**0,085**	0,842	0,840	0,838	0,877
Int	3	RF	0,850	**0,079**	0,851	0,850	0,850	0,951
Int2	2	SVM	0,900	0,115	0,901	0,900	0,899	0,892
Int2	2	IB1	0,887	0,113	0,888	0,887	0,888	0,887
Int2	2	RF	0,898	0,110	0,898	0,898	0,897	0,948
Qual2	2	SVM	0,817	0,183	0,817	0,817	0,817	0,817
Qual2	2	IB1	0,792	0,208	0,793	0,792	0,792	0,792
Qual2	2	RF	0,786	0,214	0,786	0,786	0,786	0,853
And2	2	SVM	0,900	0,114	0,901	0,900	0,900	0,893
And2	2	IB1	0,902	0,102	0,902	0,902	0,902	0,900
And2	2	RF	0,901	0,109	0,901	0,901	0,901	0,954

System Type Effect

Table 4.4 shows that metrics attained good values for several of the Systems. In order to make the differences in performance of the 4 variants of the system clear, Figure 4.12 shows the averages and respective 95% confidence intervals for all the metrics considered.

It is possible to observe that: (1) better values are obtained by the And2 and Intelligibility 2 (Int2) systems in all metrics, except false positives; (2) the worst values are obtained by the system based on Overall Quality 2; (3) Despite losing to Int2 and And2 systems in several of the metrics, the Intelligibility (Int) system (the only one with 3 classes) obtains the lowest false positive rate (around 8%).

Considering the complete hybrid NLG system, the false positive classification is the one that will cause a serious and relevant impact on the sentences to be sent, by the system, to the user. Effectively, if a sentence is erroneously marked as good and delivered to a user, it can cause serious damage, particularly if the problem is related to understandability or the quality of data supplied.

The aim of our Hybrid Data2Text system is to provide to users the best possible sentences. To make possible the replacement of a low-quality sentence, generated by the Data2Sentences module, by a better one that will be produced by Templates Module, the translation-based sentence must be assessed. The Int and And2 systems are the best options for this evaluation. With Int, as it has the lowest false positive rate, we expect to improve the intelligibility of selected sentences. With And2, which has the highest rates in all other ratios, we expect to improve the overall quality of selected sentences.

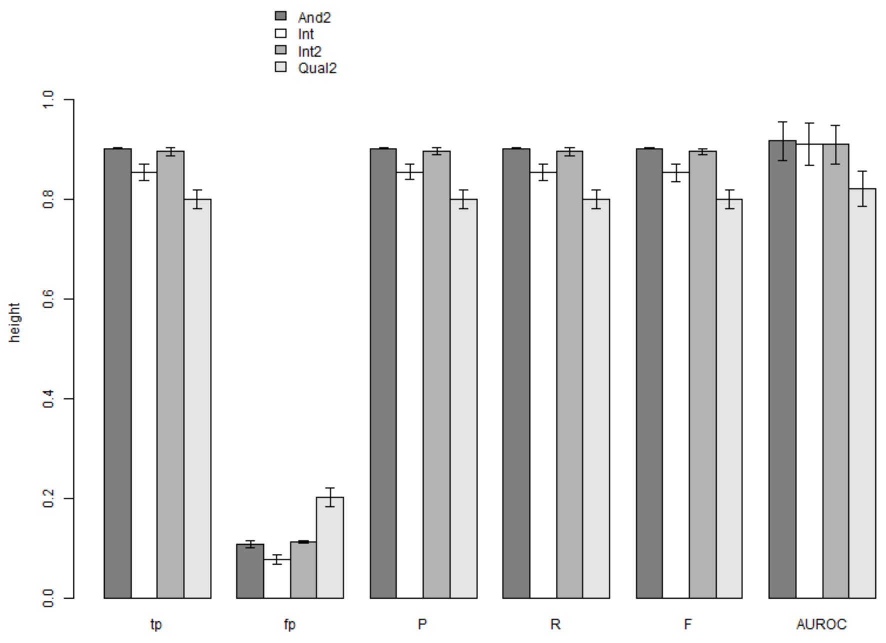


Figure 4.12: Averages and 95% confidence intervals for the metrics considered for the 4 different variants of the sentence quality evaluation module

Classifier Effect

Figure 4.13 shows the averages of evaluation metrics and their 95% confidence intervals. As presented in Table 4.4 and Figure 4.12, there is no significant difference in values obtained by the three classifiers (SVM, IB1 and RF). The ROC value obtained by the Random Forest classifier has a slightly improved value compared to SVM and IB1, but this is not statistically significant. Even on false positives – the most feared metric – the results obtained are almost the same, meaning that the choice of which classifier to use can be considered irrelevant.

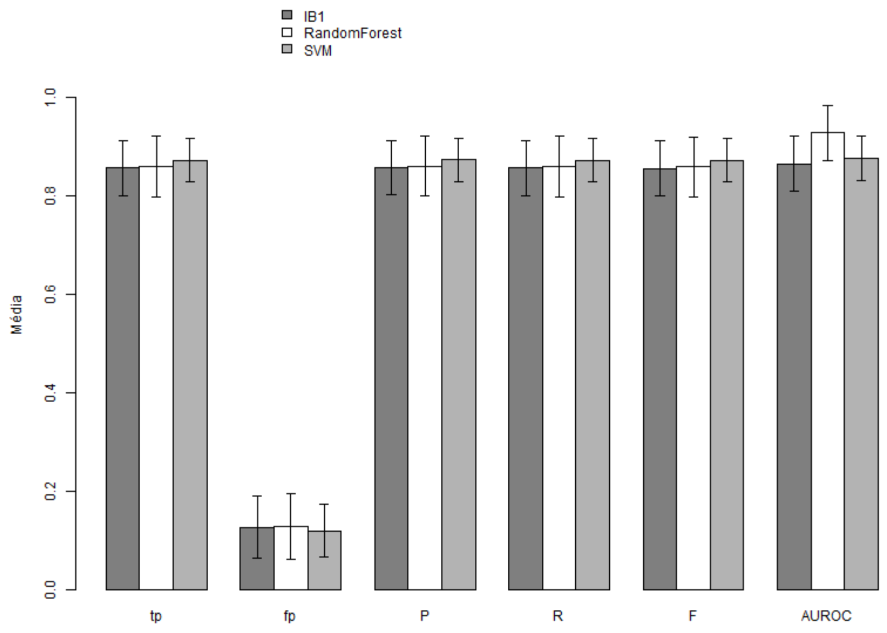


Figure 4.13: Averages and 95% confidence intervals for the metrics considered as function of the classifier

Estimate of Complete System False Positives

Despite only evaluating the sentence quality classification module, as the sentences used as the basis for the annotated corpus were the outcome of a test of the translation-based NLG module, it is possible to have a complete system estimate, at least concerning the metric which is most relevant to our system – false positives.

For Int2, we have 178 sentences marked as non-intelligible and 465 as intelligible. If this proportion is kept up by the Moses based sentence generation module, in a test with 1000 sentences, we would expect 277 non-intelligible sentences. This means that approximately 10% of them (28 sentences) would be sent to the user, resulting in a false positive rate under 3% for the complete system. A similar calculation gives an estimate of 6% (374 non-good sentences out of 643 and a `fp` rate of approximately 10%) for the complete system based on And2. The estimate for the 3 classes Int system is more complex as different false positive situations arise. If we consider as false positive only the cases which originate from sentences classified with “0” (only 49 out of 643) we obtain a result under 1%; if we also consider the intermediate class we have a situation similar to the one for Int2.

Combining the results presented in this subsection with the previous section, the two variants based on Intelligibility can be considered the best systems.

4.5 Conclusions

Driven by the growing need to transmit information from computer systems to users in spoken or written Portuguese, we presented a proposal for a hybrid NLG system, combining a translation-based generation module, a template-based module and a sentence quality evaluation module. With this system, we address the limitations caused by the inconsistent quality of translation-based generation. Sentences considered unacceptable by the sentence quality evaluation module are replaced by new ones, generated by the template-based generation module.

Several variants were implemented in the sentence quality evaluation module, differing in the type of information used for the decision-making process (Intelligibility, Quality, combination of the latter with grammaticality) and the classifiers used (SVM, IB1 and Random Forest). Evaluation with a manually annotated corpus showed a false positive rate below 8% and F-measures and ROC areas above 0.9, for the best systems. The performance of the different classifiers used did not reveal significant differences, in any of the systems evaluated. Nevertheless, systems based on the Quality quantifier showed lower classifications.

The results obtained suggest that our approach is valid. As for our system, the most relevant issue is the low false positive rate, which ensures that low quality sentences are replaced and not transmitted to users. The 8% false positive rate achieved by the best systems, while not a bad value, can be expected to be even lower in practical applications, decreasing to around 3%. We wish to remind readers that the generation module produces low quality sentences at a rate below 30%, which led to that `fp` rate.

Chapter 5

Application to another domain – Summaries2HotelManagers

We consider the work presented in the previous chapters a ‘testing ground’ of what we are aiming to prove. In this chapter, an evolution of our system is presented. We start by defining our goals and the strategy taken to define the new system. Then, we describe the production and training of the corpus. We conclude with an evaluation of the quality of produced sentences.

5.1 Domain / Scenario

5.1.1 Requirements

For the production of our new system, several concerns were taken into account. One of those was time. To build an NLG system, a huge amount of time is usually required. That was the major constraint that we faced, since we did not have much time available. Another, was the specification of what system we really wanted to build.

We are particularly interested in finding a system that combines multimodal interaction with our team’s ongoing projects. The aim was to find a system that produces good quality utterances with high variability. The complexity of this new system is another concern, as it should be more complex than the first one, but still be aligned with state-of-the-art data2text systems. Ideally, it should be in a new, little explored domain.

Our process was quite simple. First, we produced a list of about fifteen possible application ideas. Then, we review each one against the rules explained above, and narrowed the list down to five. The candidates on this short list were: a canteen information system, used to 'talk' about the menu of that canteen; a system to summarize timetables of a railway system; a system to report the scores of several sports; a system to report movie reviews; and lastly, a system to report and summarize evaluations made by hotel customers.

After analyzing these options, we chose the last one. The goal of this new system is to give hotel managers a tool which summarizes customer reviews. Firstly, through individual summaries for each type of service. And secondly, a general summary, by hotel or dates.

5.1.2 Hotel Scenario

The proposed work, which we named SUMMARIES2HOTELMANAGERS, is integrated in an ongoing project, related to data extraction. The aim of that project is to collect and summarize data about ratings of hotel services, given by their customers. The data provided consists, mainly, of hotels names, names of services, classifications given to services and the number of people who classified those services.

Our expectation, when the system is complete, is that it should be able to produce a short text which sums up the opinions that customers have expressed about a given hotel's services. This text could be sent to the hotel's director by email, at regular intervals or certain times, for instance, daily, every morning, or at the start of the week.

Figure 5.1 shows a sample email, which might be sent to a hotel director. Figure 5.2 shows several examples of the kind of sentences that we expect our system to produce. In both cases, is possible to observe the word AND (in English) or E (in Portuguese) in uppercase. This is only to indicate that that sentence was made from two different but related sentences.

5.1.3 Timeline

As described before, one of our goals is to create a system from limited resources. Time is a very valuable resource. The definition and creation of SUMMARIES2HOTEL MANAGERS took about 110 hours, which is the equivalent of 3 weeks' work. If we take into consideration the time it took to acquire the seeds of the corpus, the time spent exceeded 6 weeks. Table 5.1 shows the major tasks which were performed, and the *in English:*

Email to *Operations' Director* of Mélia Aveiro: 24 de April 2017

Subjet: Month evaluations

Your hotel had a mixed review in restaurant service AND
bad at room service. The price is high.

Your competitor, XX, had good evaluation at restaurant service AND
very good at room service.

Their price was considered as good.

Automatically generated by AIE + NLG

in Portuguese:

Email para Director de Operações do Mélia Aveiro: 24 de April 2017

Assunto: avaliações do mês

O seu Hotel teve avaliação mista nos serviços de restauração E
péssima nos quartos. O preço é alto.

O seu concorrente, XX, teve avaliação boa nos serviços de restauração E
muito boas nos quartos.

O preço dele foi classificado como bom.

Gerado automaticamente por AIE + NLG

Figure 5.1: Example of a possible email containing a hotel's evaluation

in English:

The hotel [Name] has good rooms.

Rooms have obtained very good evaluations.

The [3 stars] ALIF [address] had mixed review at restaurant services AND
very bad at room service.

The breakfast at ALIF received several bad reviews.

The 5 stars Mélia, at José Malhoa street, has a very good breakfast.

in Portuguese:

O Hotel [Nome] tem quartos bons.

Os quartos obtiveram classificações muito positivas.

O [3 estrelas] ALIF [no Localização] teve avaliação mista nos serviços de restauração E
péssima nos quartos.

O pequeno-almoço do ALIF foi alvo de críticas.

O 5 * Mélia, na José Malhoa, tem pequeno-almoço excelente.

Figure 5.2: Example of possible sentences to be generated by our system

amount of time, in hours, required to complete them. This information was collected using the Toggl[1] tool.

Table 5.1: Main tasks and their duration.

Task description	Duration
Create web site to collect corpus	34h 09min
Collect corpus (phase 1)	08 days
Collect corpus (phase 2)	10 days
Prepare corpus for expansion	18h 41min
Corpus expansion	07h 48min
Rectify Portuguese syntax and grammar from expanded corpus	31h 18min
Prepare IE + OD1 + OD2 corpus for LM	06h 28min
Train and improve corpus	11h 41min

5.2 Corpora and training

5.2.1 Corpus structure

As in our first system, our corpus has two perfectly aligned languages. Once again, we assigned one language as 'input language' and the other as 'output language'. The input language is made of data extracted from a database. Table 5.2 presents the tokens used for this language.

Table 5.2: *Tokens* of SUMMARIES2HOTELMANAGERS and their correspondences.

Token	Correspondence
Hotel name	name of hotel, being evaluated
Service name	name of hotel's service, that is going to be evaluated
Evaluation	evaluation to service
Number people	among of people who did the evaluation
First adjective	principal adjective that reflects evaluation
Second adjective	secondary adjective

The first token corresponds to the hotel's name. The second token is related to name of the specific hotel service being evaluated. The Evaluation token expresses the

[1]https://toggl.com/

service's evaluation, in three possible values: positive, negative or neutral evaluation. The fourth token defines the number of people that made the evaluation. It is configured with five possible value options: very few people, some people, half of evaluators, many people, almost all people. The fifth and sixth tokens are related to adjectives used to evaluate the hotel's services. In all tokens, data is compressed and white spaces are replaced by hyphens. That is done to make MOSES' work easier, because that way it treats token names which consist of several words as a single word. When there is no data, the token is filled with the word 'sem-valor' (no-data).

Table 5.3 shows a sample of our corpus. These sentences are from the A system (see Table 5.4).

Table 5.3: Sample of aligned corpus (six first lines of training corpus A).

Linguagem interna	Frase correspondente
hotel-sem-valor servico-atendimento aval-neutra algumas-pessoas adj-diferente adj-aceitavel	algumas pessoas classificaram de forma neutra o atendimento, como diferente dos outros hotéis e aceitável
hotel-sem-valor servico-funcionarios aval-neutra metade-pessoas adj-sem-valor adj-sem-valor	Parte dos clientes não se pronunciaram quanto aos empregados
hotel-apart-boa-vista servico-restaurante aval-positiva muitas-pessoas adj-fantastico adj-cuidado	muitos cliente gostaram do restaurante dos Apartamentos-Boa-Vista por ser cuidado e fantástico
hotel-faro-vintage-guest-house servico-restaurante aval-positiva poucas-pessoas adj-fantastico adj-sem-valor	Algumas pessoas acharam o restaurante do Faro-Vintage-Guest-House fantástico
hotel-cr-correeiros servico-quarto aval-negativa algumas-pessoas adj-isolamento adj-barulhento	motivadas pelo barulho e pelo isolamento, o quarto foi classificado negativamente por algumas pessoas, no CR-Correeiros
hotel-sem-valor servico-pessoal aval-positiva poucas-pessoas adj-acessivel adj-educacao	poucas pessoas classificaram positivamente o pessoal do hotel, apesar de serem acessíveis e educados

5.2.2 Corpus preparation

The first task was to study the available data and define the tokens which are going to make up the input language. Next, about 20 people, aged 20 to 60 years old, produced sentences summing up the data contained in the tokens. This task was done in two stages. In the first, we collected 135 sentences. After that, a second set of 230 sentences was collected. In the next task, both sets were used to produce an expanded corpus, as described in Section 3.2.2.

To evaluate our system, several combinations of corpora and language models were made. These combinations are presented in Table 5.4. First, we named the first 135 seed sentences *corpus A*. The second seeds were not used because we identified several problems in them that do not invalidate their use as seed, but invalidate their use *per se*. This corpus was named as the *original corpus*.

Secondly, for the *B corpus* we extended the A corpus with all token data used to describe the *input* language. This corpus is referred to as the *extended corpus*.

A third, *C corpus* was created, expanding the B corpus with all sentences produced by applying both sets of seed sentences. The process of extension was as described before. This corpus is referred to as the *expanded corpus*.

As *IE* we named the sentences produced by hotel customers when evaluating hotels. These sentences are in the same domain as sentences produced by the system in evaluation. The *IE corpus* was only used to enrich the corpus used to produce the *language model*, and consists of about 129 thousand sentences.

To enrich the language model, two other corpora were collected. The *OD1 corpus* was made from Portuguese minutes from the European Parliament, available at (Rosas et al., 2014, Machado et al., 2010). This minutes are part of the Moses for Mere Mortals (MMM) system, a prototype which aims to help the production of a translation chain for real world documents. These minutes are made up of 200 thousand sentences. The final corpus, *OD2*, consists of Portuguese sentences obtained from the Público newspaper. These sentences belong to the CETEMPúblico corpus (Santos and Rocha, 2001, Rocha and Santos, 2016) created by Linguateca project. *OD2 corpus* has about 860 thousand sentences. Both corpora are *out of domain* in relation to our system. Our goal is to enrich produced sentences.

5.2.3 Samples

Table 5.5 shows several samples of sentences produced by SUMMARIES2 HOTEL-MANAGERS. These sentences were produced by systems trained with different corpora, as described in Table 5.4.

The first three samples presented in this table show distinct sentences produced

Table 5.4: Corpora used in this evaluation.

Corpus	Sentences	Description
A	135 sentences	human made sentences, in-domain
B	435 sentences	A extended with input language tokens
C	2.781 sentences	B expanded with new sentences obtained from seeds
IE	129.130 sentences	corpus (in-domain) for language model
OD1	+200.000 sentences	out-of-domain – minutes from European Parliament
OD2	+860.000 sentences	out-of-domain – CETEMPúblico

by the system. Each pair consists of the *input* sentence and the corresponding *output* (generated) sentence. All sentences were produced by a system trained with the C + IE + OD1 + OD2 corpora (see Table 5.4). It is possible to see that each of the sentences generated has a different level of quality. The first is considered 'good' as it is grammatically correct and transmits all the data expressed in the *input* sentence correctly. The second sentence is considered 'acceptable' as it transmits the general idea of what the *input* sentence expresses, but it is grammatically incorrect and does not convey all of the data. The last sentence (number 3) is considered 'wrong' since it is not at all comprehensible.

The second part of the table (sentences 4 to 7) shows an alignment between the sentences produced by humans and sentences produced by the system, for the same input value. Sentences are presented in lower case so as to show the differences between human and system sentences. For each data input, when a sentence is produced by the system, it is usually different from how a human would write it. These differences can be the adding of new words, erasing words, or by changing their position within the sentence. Here, as before, we use the token "***" to highlight the adding or erasing of words. Words in upper case are used to show where words have changed position in the sentence. This is used in the human produced sentences only to emphasize the differences between the two sentences.

Samples 6 and 7 show sentences produced by the system. They are good, in terms of naturalness and intelligibility. They are totally different from those produced by humans, but nevertheless, express the same data as their corresponding human sentences. As observed, when we evaluated our first system, these sentences are, usually, marked "wrong" by automatic evaluation tools. This renewed our resolve to carry out a human evaluation of the sentences produced by SUMMARIES2HOTELMANAGERS, as well.

In the third and final part of table, three sets of sentences generated by different

Table 5.5: Samples of sentences produced by SUMMARIES2HOTELMANAGERS.

	Samples from the input language and corresponding generated sentence
#	Sample
1	*input*: hotel-sem-valor servico-jantar aval-positiva todas-pessoas adj-bom adj-sem-valor *output*: o jantar foi bom para a esmagadora maioria das pessoas
2	*input*: hotel-sem-valor servico-atendimento aval-negativa poucas-pessoas adj-pessimo adj-sem-valor *output*: atendimento péssimo serviço
3	*input*: hotel-sem-valor servico-hotel aval-neutra algumas-pessoas adj-sem-valor adj-sem-valor *output*: o hotel , algumas pessoas
	Samples of produced sentences (S) aligned with the corresponding sentence made by humans (H)
4	*input*: hotel-sem-valor servico-atendimento aval-positiva sem-valor-pessoas adj-agradavel adj-impecavel H: NESTE HOTEL, o atendimento era agradável e BASTANTE impecável S: *** *** o atendimento era agradável e *** impecável
5	*input*: hotel-canico-seaside-apart servico-cheiro aval-neutra sem-valor-pessoas adj-normal adj-sem-valor H: NO CANICO-SEASIDE-APARTMENT, o cheiro é DESCRITO COMO normal S: *** *** o cheiro DO CANICO-SEASIDE-APARTMENT é normal
	Samples of generated sentences, very different from test sentence, but with quality
6	*input*: hotel-sem-valor servico-barulho aval-neutra todas-pessoas adj-aceitavel adj-sem-valor H: OUVIA-SE ALGUM BARULHO DE FORA MAS NÃO ERA MUITO INCOMODATIVO S: A ESMAGADORA MAIORIA DAS PESSOAS DESCREVEM O BARULHO COMO ACEITÁVEL
7	*input*: hotel-sem-valor servico-mobiliario aval-positiva poucas-pessoas adj-bonito adj-funcional H: POR ESTAR BONITO E SER FUNCIONAL, DIVERSOS CLIENTES AVALIARAM O MOBILIÁRIO POSITIVAMENTE S: *** *** *** *** DIVERSOS HÓSPEDES GOSTARAM DO MOBILIÁRIO BONITO E FUNCIONAL
	Sentences produced from different sub-systems, with the same input sentence
8	*input*: hotel-sem-valor servico-rececao aval-positiva metade-pessoas adj-charmosa adj-sem-valor C: a receção cerca de metade dos clientes , como charmosa C+IE: a receção , cerca de metade dos clientes charmosa C+IE+OD1+OD2: a receção é charmosa
9	*input*: hotel-sem-valor servico-disponibilidade aval-positiva muitas-pessoas adj-grandioso adj-sem-valor C: muitas pessoas classificaram positivamente a disponibilidade do hotel é grandioso C+IE: muitas pessoas classificaram positivamente a disponibilidade do hotel é grandiosa C+IE+OD1+OD2: a disponibilidade é grandiosa , muitas pessoas
10	*input*: hotel-sem-valor servico-estacionamento aval-positiva sem-valor-pessoas adj-bom adj-excelente C: o estacionamento é bom e excelente C+IE: o estacionamento é bom e excelente C+IE+OD1+OD2: o estacionamento é bom e excelente

systems are shown. Each set has the same *input* sentence so that it is possible to compare *output* sentences. As described previously, the systems were trained with the C, C + IE and C + IE + OD1 + OD2 corpora (see Table 5.4). In the first set (sample 8), every *output* sentence was considered 'acceptable'. They are very different, expressing the same data in different ways. The last sentence is the only one which is grammatically correct. In the second set (sample 9), the first and second sentences are quite similar. The difference between them is quite subtle, consisting only of the last word. The last letter, to be precise. Both sentences are 'good', but the second is the best, because its last word agrees with gender of the sentence. The last set (sample 10) shows three *output* sentences that are exactly the same, although they were produced by different systems.

5.3 System Evaluation

In this section, we present several samples of sentences produced by the SUMMARIES2 HOTELMANAGERS system. Later on, a formal evaluation carried out by automatic tools is presented. Finally, we present the results of a human evaluation of the sentences produced by this system.

5.3.1 Quantitative evaluation

Evaluation method

We produced several versions of the SUMMARIES2HOTELMANAGERS system, with the corpora described in Table 5.4. Our intention was to test the influence of corpus expansion and the use of several corpora in the production of language models. To test and train all 'sub-systems', we used the *10-fold cross-validation* technique, as explained in the *Corpus preparation for training and testing* section (page 58). After training, all systems were tested with the corresponding test corpus. The BLEU (Papineni et al., 2002) and Meteor (Denkowski and Lavie, 2014, Denkowski and Lavie, 2011) metrics were used to measure the accuracy of each system.

Due to the use of *10-fold cross-validation* training, for each 'sub-system' 10 different versions of it were produced. We named each of these trained systems with a letter from A to J (not to be confused with the letters in Table 5.4).

5.3.2 Effect of corpus expansion

Our first evaluation concerns the effect of corpus expansion. It is well known that MOSES works well with a corpus of thousands/millions of sentences. Our initial corpus only has 135 sentences. How did it perform? What was the influence of the extension with token data on the *input* language? Is the expansion of the original corpus with new sentences, obtained as explained in Section 5.2.2, beneficial? These are the questions that we have tried to answer. Figure 5.3 shows the results of the BLEU evaluation of the corpora named *ORIGINAL* (the A corpus), *EXTENDED* (the B corpus) and

EXPANDED (the C corpus) (see Table 5.4).

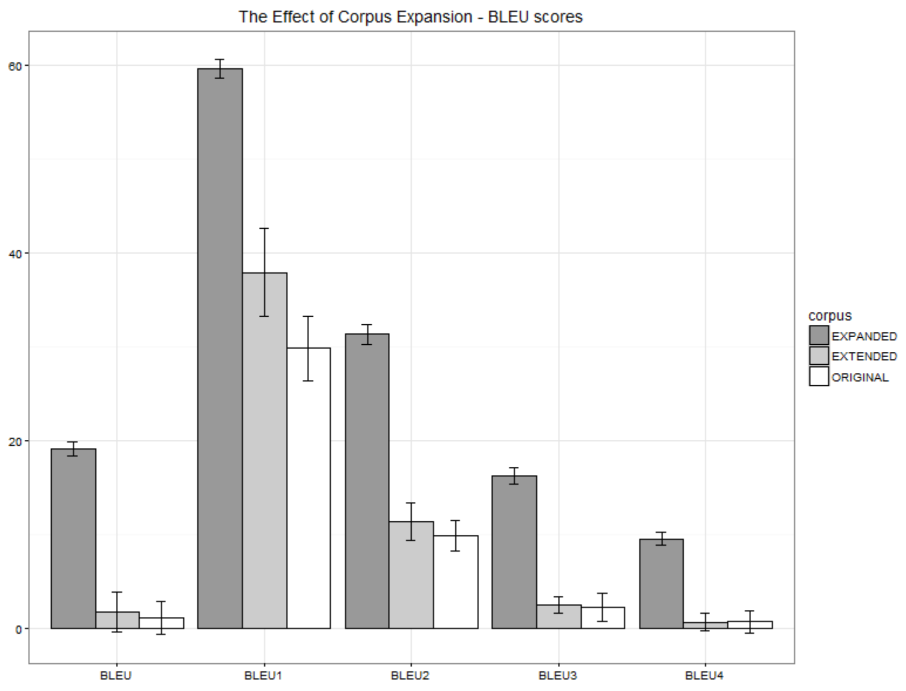


Figure 5.3: BLEU evaluation of the original, extended and expanded corpus

BLEU evaluates the strict correspondence between the evaluated corpus and the reference corpus. Bearing this in mind, the BLEU global score result obtained by the *Original* corpus is not surprising. As expected, the improvement of the original corpus using tokens' data (*B* corpus) provided better overall results. The highest improvement was obtained at Bleu1. Effectively, at uni-gram level the correctness of sentences provided by the B corpus are better than sentences provided by the A corpus. This advantage starts to decrease at Bleu2, to a statistically insignificant level, once the *confidence intervals* start to intersect each other.

The expansion of the B corpus with new sentences (*C corpus*) resulted in a better performance at all levels of the BLEU evaluation. The improvement was quite marked. BLEU scores corpora from 1 (total correspondence between the evaluated corpus and the reference corpus) to 0 (no correspondence). The scores obtained by the C corpus do not mean that its sentences are, globally, not good. It only means that the generated sentences are not aligned with the reference corpus, which is not necessarily bad. As shown in the middle section of Table 5.5, sentences can differ from reference sentences and still be good.

Figure 5.4 shows the Meteor evaluation of the same corpora tested with BLEU. Here, it is evident the best evaluation results were obtained by the C corpus, in all metrics. For the F1 metric, which measures the relation between precision and recall at uni-gram level, the C corpus scored almost twice as high as the B corpus. Almost the same result is obtained with F_{mean} (which favors *recall*) with these two corpora. The differences, with these two metrics, are also statistically significant between the B and A corpora, though not as large as the difference between the C and B corpora.

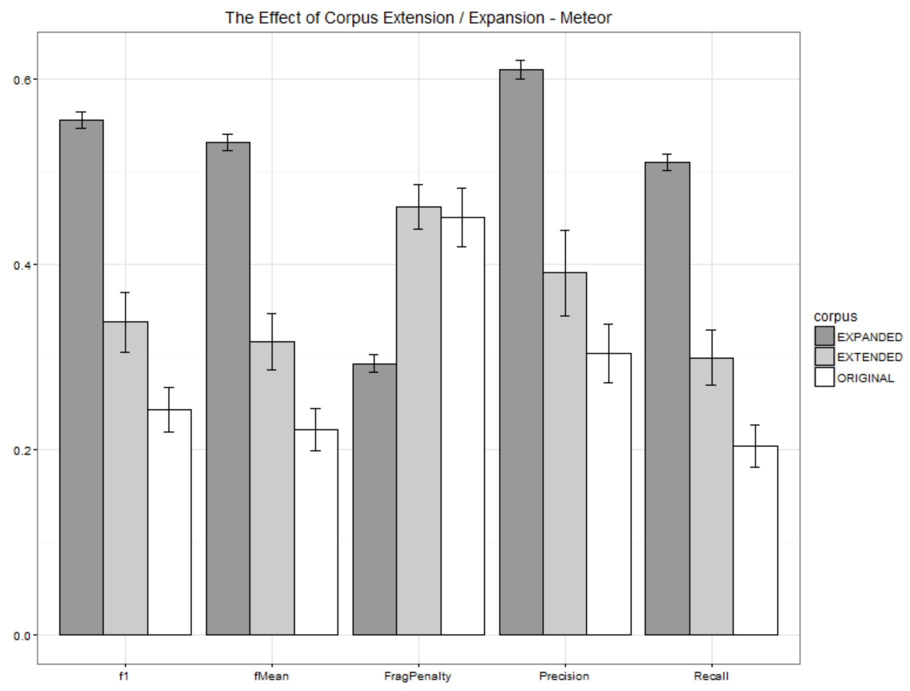


Figure 5.4: Meteor evaluation of the original, extended and expanded corpus

The same relationship is observed with the *Precision* and *Recall* metrics, for all corpora. For *Fragmentation Penalty*, results do not follow that tendency. As expected, the C corpus has a lower penalty because it is richer. For the A and B corpus, the difference is not statistically significant, although the B corpus has a slightly higher penalty. That is probably because token data, which only consists of uni and bi-grams, was added to the B corpus.

5.3.3 In-domain corpus effect on Language Model

In short, Language Model (LM) is a tool used to produce new sentences, and each new sentence must be syntactically and morphologically correct. It is common knowledge that every language follows rules. Some of them have many complex rules. In those cases, modeling all the rules of a language is a very hard task. The Portuguese language is one such case, and its richness makes this task harder still.

This LM tool is an automatic tool used to 'learn' the rules of a language. Manning, et al. define Language Model as "*a function that puts a probability measure over strings drawn from some vocabulary*" (Manning et al., 2008, page 237) – i.e., it statistically analyzes the relation between the words present in sentences, of a certain language. Because words do not combine in a random order, the presence of n-grams establish the probability of one, two or more words being part of a sentence, in a particular combination. These probabilities are used to estimate, and then produce, new sentences.

What was the influence of the corpus on the production of a valid LM for Portuguese, connected to our system? Figure 5.5 and Figure 5.6 show the results obtained by our system, trained with the B corpus and two different language models. Language models should be produced with a corpus based on the output language. Accordingly, for the first version, we used only the output language as seed for the language model production. For a second version, we used the output language and information extracted (IE) from a system of hotel analysis by hotel costumers, as the seed for the language model.

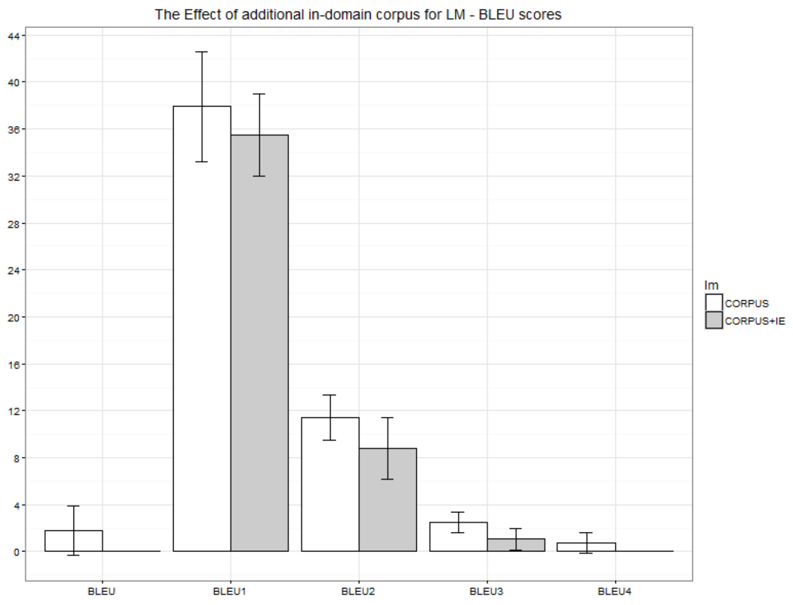


Figure 5.5: Influence of different LM – BLEU evaluation

In both evaluations, the scores obtained by the system trained with only the output corpus as seed to the language model are better than the scores obtained by the system which was trained with output language plus a corpus with extracted information as seed for the language model. Again, this does not mean that the second version of the system produces worse sentences than the first system. Table 5.6 shows two samples of sentences produced by these two systems, for the same input. In both samples, the

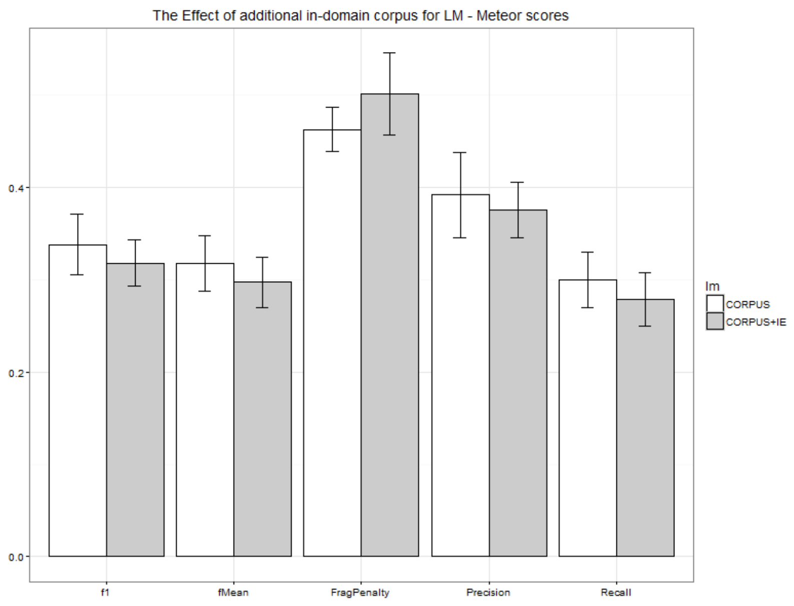


Figure 5.6: Influence of different LM – Meteor evaluation

---

sentences produced by the system with the extended language model were considered better in an informal human evaluation.

Table 5.6: Output samples of systems, based on B corpus, trained with different LM.

input 1	hotel-sem-valor servico-preco aval-positiva poucas-pessoas adj-acessivel adj-sem-valor
B	os clientes do hotel a qualidade de serviço acessível
B + IE	a qualidade do hotel é acessível
input 2	hotel-sem-valor servico-wc aval-negativa muitas-pessoas adj-desconfortavel adj-sem-valor
B	o wc a hotel é desconfortável
B + IE	o wc é muito desconfortável para o hotel

Figure 5.7 shows the scores obtained in the Meteor evaluation of two systems trained with the C corpus and the same variations in the language model as described in the pages above. Scores obtained by the system with an improved language model are slightly worse than scores obtained by the simpler system. As observed above, that does not necessarily mean that this system performs worse. Like Table 5.6, Table 5.7 shows two samples of output sentences produced by these two systems, for the same input.

In some cases, the differences are quite minor. In others, they are substantial. For our goals, this is not a problem, as we are actively trying to design a system which can produce more creative and varied sentences.

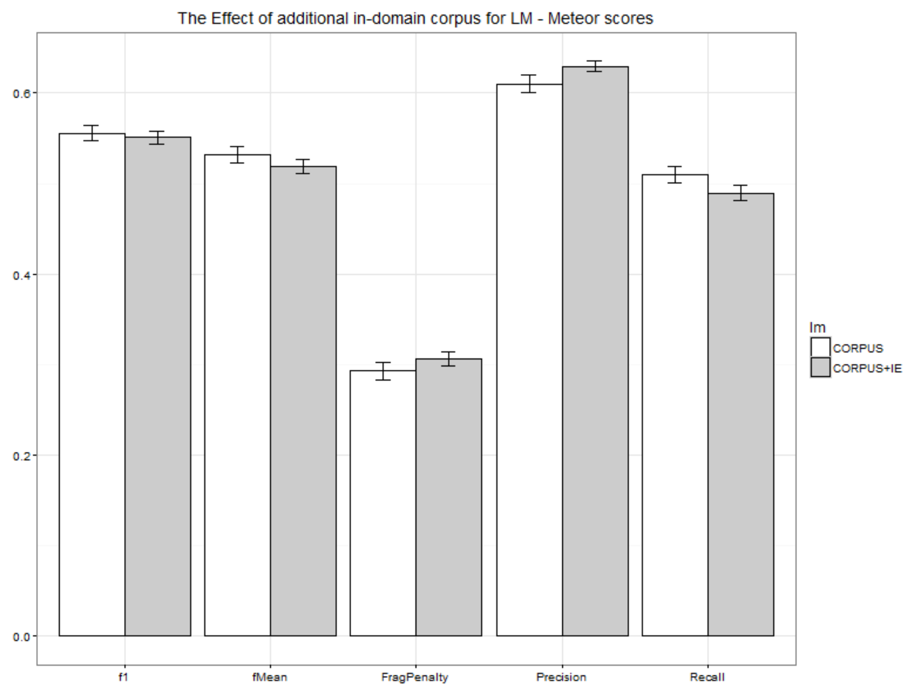


Figure 5.7: Influence of different LM, on training with C corpus – Meteor evaluation

Table 5.7: Output samples of systems, based on C corpus, trained with different LM.

input 1	hotel-sem-valor servico-restaurante aval-positiva todas-pessoas adj-bom adj-atencioso
C	o restaurante , a esmagadora maioria dos clientes é bom , sendo descrito como atencioso
C + IE	o restaurante a esmagadora maioria das pessoas consideram bom e atencioso
input 2	hotel-sem-valor servico-profissionalismo aval-positiva sem-valor-pessoas adj-excelencia adj-responsavel
C	o profissionalismo era excelência e bastante responsável
C + IE	o profissionalismo é responsável de excelência

5.3.4 Out-of-domain effect on the Language Model

In this final section, we show the influence that a richer corpus can have on the production of the language model. That enrichment was achieved by the addition of a corpus, with sentences from the same domain as the output corpus. This new experience, tries to study the influence of an out-of-domain corpus on the language model.

Here, we are going to use two different corpora. Firstly, a set of minutes of European Parliament in Portuguese – referred to as the *OD1 corpus*. Second, a set of sentences in Portuguese obtained from the 'Público' newspaper, and belonging to the Linguateca project – referred to as the *OD2 corpus*. We are going to use the C corpus as the training corpus, because previous experiences showed that this is the one which achieves the best results.

Figure 5.8 shows the results obtained by systems trained with three different language models. The first language model was obtained by the output corpus plus the in-domain corpus (as explained in the previous section). For the second language model, we added, to previous LM, the OD1 corpus. Lastly, for the third language model, we added the OD1 and OD2 corpora.

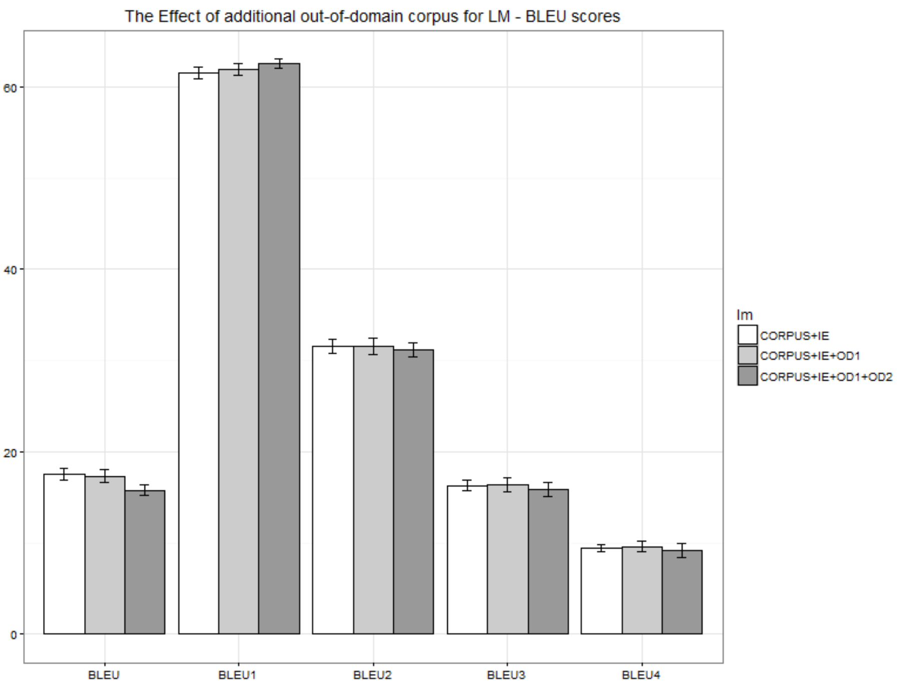


Figure 5.8: Out-of-domain effect on LM – BLEU evaluation.

The differences between the metrics (Bleu, Bleu1 to Bleu4) are not statistically significant. Nevertheless, a slight improvement is observed in the Bleu1 metric.

The Meteor evaluation (Figure 5.9) reveals that there are no significant differences from what was observed in the BLEU evaluation. However, it is possible to observe that the addition of an out-of-domain corpus to the language model increases the *precision* metric, with a consequent reduction of the *recall* values. This means that translated sentences are closer to the reference sentences, than sentences translated with the system with the simpler language model.

As automatic metrics only express the mathematical difference between translated sentences and reference sentences, it is difficult to be sure of the correctness of translated sentences, by this method alone. Of course it is a good reference point, but human languages are quite rich and that is difficult to assess with automatic metrics.

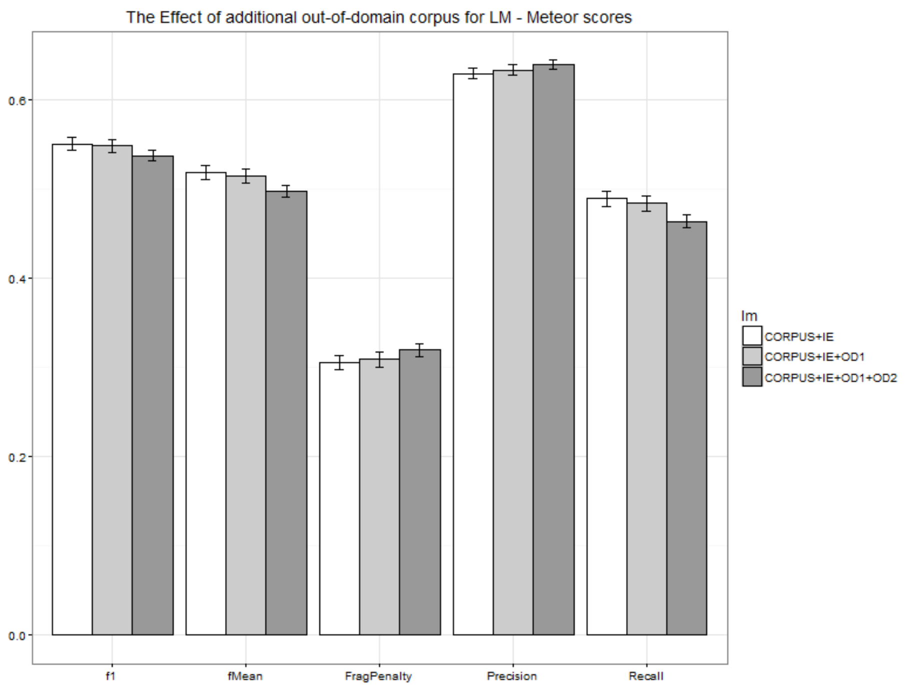


Figure 5.9: Out-of-domain effect on LM – Meteor evaluation.

5.3.5 Comparison with a related system

Previous sections presented the evaluation results obtained by different variations of our system. The results obtained through automatic evaluations are consistent. Better results were obtained by the C corpus with variations of the language models. Figure 5.10 compares the results from our best systems with Mountain's best systems (Langner, 2010, p. 73–75).

For this comparison, we chose the system obtained from the C corpus, with two language model variations: the language model made only with the output language, and the language model made with the output language plus IE and the OD1 and OD2 corpora. From Mountain, we chose the original system and the system with the best results. For clarity, only the Meteor evaluation is being presented, since the results from BLEU are similar. In all metrics, our system's results are better. At *F1* and *Recall* level, results are almost twice the ones obtained by MOUNTAIN systems.

A similar comparison was made before, between our first system and MOUNTAIN (see Session 3.4). Results obtained by SUMMARIES2HOTELMANAGERS showed it to be better than MOUNTAIN, but slightly worse than our first system. This drop in performance, between our systems, does not necessarily mean that the SUMMARIES2HOTELMANAGERS system is worse than our first one (MEDICATION2PT). As was previously mentioned, if the sentences produced are different from the reference

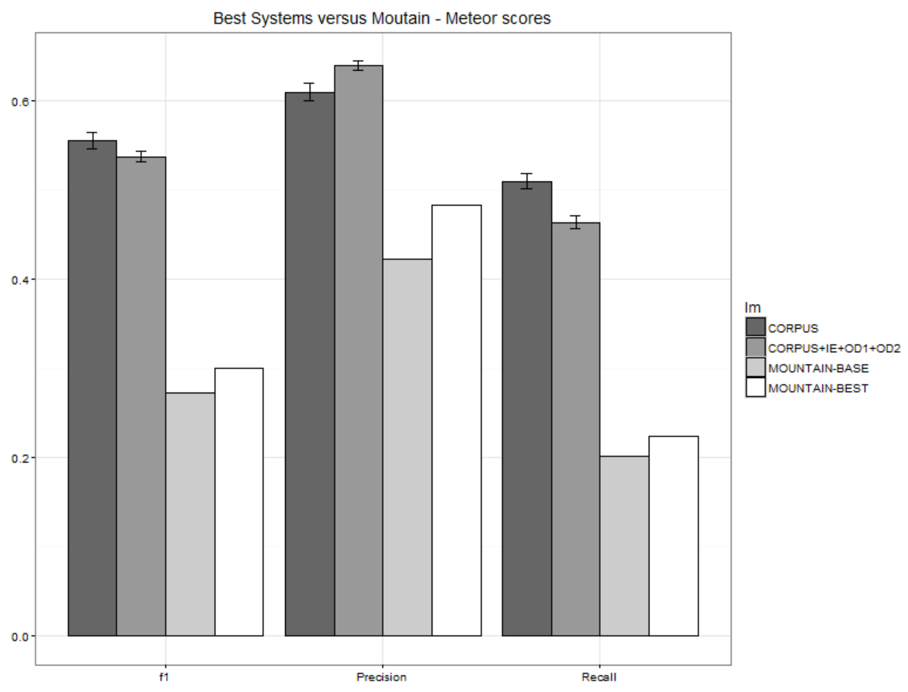


Figure 5.10: Our best systems versus MOUNTAIN – Meteor evaluation

sentences, the automatic metrics decrease their value, and that was often the case in this last system.

5.4 Human Evaluation Results

5.4.1 Method

Despite the overall high quality obtained by automatic metrics, an informal human evaluation of the produced sentences showed the quality to be unreliable. Some were quite good. Some were not. As with the first system, we decided to carry out a human evaluation to assess quality from the perspective of human users. After all, the sentences produced by our system are intended to be used by them.

Automatic evaluations suggests that systems trained with the C corpus produce better results. For this reason, we chose sentences from systems trained with that corpus. Three systems were selected. The differences between them are at the level of the language model: LM obtained only from the output corpus; LM made out of the output corpus and the in-domain corpus; and, lastly, LM produced with all available corpora (output + IE + OD1 + OD2).

As previously, all systems were trained with the 10-fold cross-validation technique.

Therefore, in order to be equally distributed, all sentences were obtained from same trained system. We randomly selected 30 sentences from each subsystem. And, from each subset of 30 sentences, we randomly chose 3 sentences, and duplicated them, to test if evaluators were consistent in their evaluation. At the end of this process, we had a set of 99 sentences.

This entire set was randomly ordered, and two aspects of each sentence were evaluated. The first one, is whether all the data meant to be included in sentence is present. Secondly, whether every sentence was syntactically and grammatically correct, even if it does not represent the data. Table 5.8 is shows the questions put and response options given to evaluators.

Table 5.8: Human evaluation – questions and options presented to human evaluators of SUMMARIES2HOTELMANAGERS.

Questions
Sentence express correctly all data?
Sentence was syntactic and grammatically correct?
Options
5 = very well
4 = well
3 = not good, neither bad
2 = almost bad
1 = bad

5.4.2 Evaluators

A total of 15 people participated in the evaluation, 6 female and 9 male, all native speakers of Portuguese. Their ages ranged from 25 to 60 years old, and the came from different professional backgrounds (students, teachers, engineers, etc.) and levels of education (high school to college). Some of them had taken part in the human evaluation of our first system.

The consistency of the evaluators was verified by repeating a few sentences (Rosenthal and Rosnow, 1991, p. 47) – specifically, 3 of the sentences for each of the 3 variants were randomly selected and repeated. Rank correlation and percentage of agreement between the 2 scores (*data* expressed in the sentence; *quality* of the sentence) was calculated for each repeated sentence. The results are shown in Table 5.9.

Despite the existence of low values, many of the cases had high correlations and percentages of agreement. To evaluate if there was agreement among evaluators, *rank correlation* and p*ercentage of agreement in attributed scores* was adopted. Rank corre-

Table 5.9: Correlation and % of Agreement between sentence and repeated sentence

Sentence ID	Repeated Sentence ID	Correlation (data)	Correlation (quality)	% Agreement (data)	% Agreement (quality)
32	20	0.70	0.37	53	33
33	30	0.55	0.23	73	47
31	10	1.00	-0.10	87	80
65	53	0.25	0.52	60	73
66	63	0.60	0.68	80	53
64	43	1.00	1.00	100	100
99	96	0.74	0.57	53	60
98	86	0.73	0.28	53	53
97	76	0.85	0.83	80	80

---

lation was calculated using kendall in R. Results obtained for 'data in the sentence' and 'quality of generated sentences' are represented in a graphical format in Figure 5.11. It is possible to observe that there are small discrepancies between evaluators in data evaluation, since not all evaluators evaluate sentences in the same way. In the second evaluation – on the quality of the sentences – almost all evaluations match up.

The average value obtained was 0,40 and 0,61 for quality. All the correlations were positive (see Figure 5.11). The confidence value for the set of n speakers (n=15), named 'effective confidence', was calculated using Spearman-Brown formula (Rosenthal and Rosnow, 1991, p. 51):

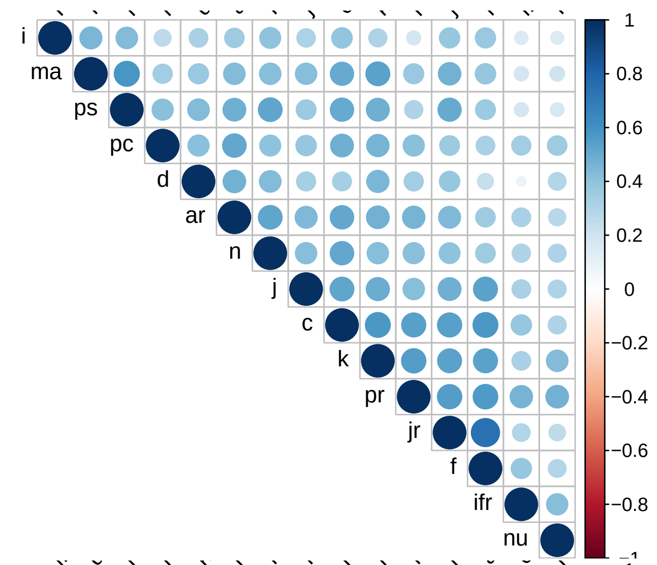
$$R = \frac{nr}{1 + (n-1)r}$$

The value obtained for R was 0,91 and 0,96 for data in the sentence and sentence quality, respectively.

Combining the information above, the results obtained in judge-to-judge correlation and consistency in scores for the same sentence did not cast serious doubts on the validity of the results.

5.4.3 Results

The following figures show the results of the evaluation of the sentences. All figures show the average of scores obtained, with a confidence interval of 95%. Figure 5.12 shows the results concerning how data is expressed in the sentences. Figure 5.13 shows how good the sentences are, grammatically and semantically. In both cases, the figure on the right side of the page is a zoomed version of the figure on the left side of the



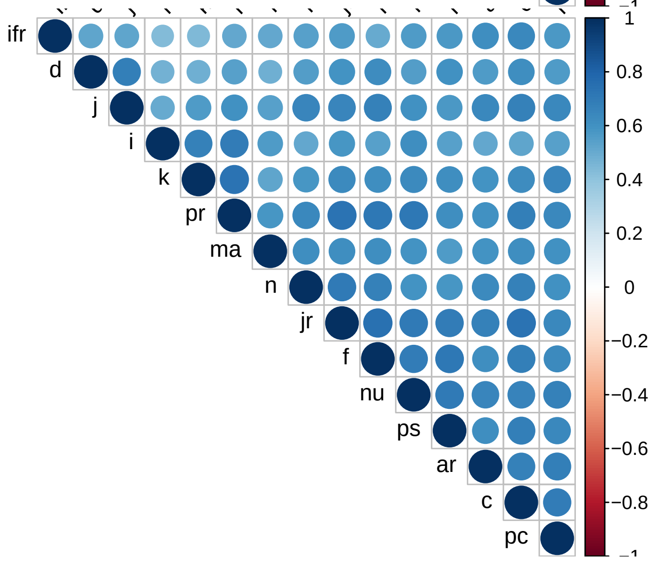


Figure 5.11: Correlation between evaluators
top: data on sentence evaluation; *bottom*: quality of sentence

page.

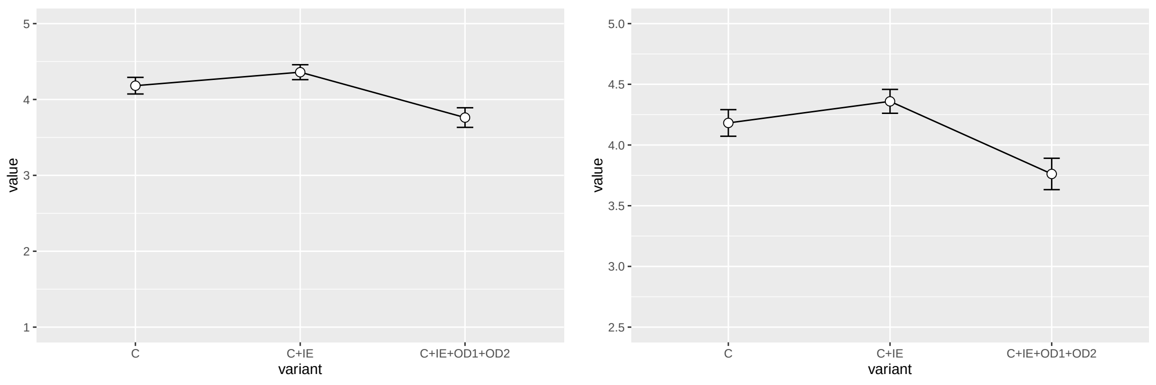


Figure 5.12: Evaluation of how data is expressed in the sentences

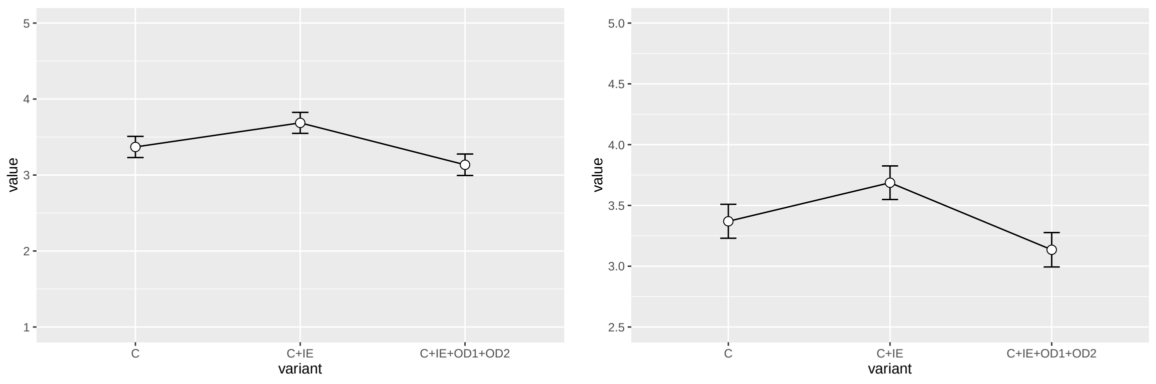


Figure 5.13: Evaluation of sentence quality (grammar and semantics)

Scores obtained show that the combination of the C corpus with the language model expanded with sentences obtained from the domain of hotels obtain better results than other systems. We can observe that the addition of out-of-domain sentences to the language model, in this case, resulted in worse evaluations (see Table 5.10). Evaluators expressed that the sentences are better at expressing data (best value – 4.36, for the C+IE system) than in terms of quality (best value – 3.69, for the C+IE system).

To get a better insight on the differences between systems, we investigated the distribution of scores. Figures 5.14 to 5.19 show the scores obtained by the three systems, for the two questions being evaluated. Figure 5.14 and Figure 5.15, as well as Figure 5.16 and Figure 5.17, express different views of the same data, showing the relation between the number of sentences in each system and the score that they achieved. Figures 5.18 and 5.19 present the same data, with a focus on systems rather than on scores.

In the evaluation of the data expressed in the sentences, systems C and C+IE had scores between 3 and 5 in over 90% of the evaluations. System C+IE+OD1+OD2 had

Table 5.10: Evaluation results.

Scores of evaluation of how data are expressed on sentences

Variant	Value	Standard Deviation
C	4.18	1.24
C+IE	**4.36**	1.11
C+IE+OD1+OD2	3.76	1.46

Scores of evaluation of quality of sentences

Variant	Value	Standard Deviation
C	3.37	1.58
C+IE	**3.69**	1.56
C+IE+OD1+OD2	3.14	1.60

a slightly lower result. Only 79.4% of evaluations were scored 3, 4 or 5. The best system (C+IE) had 62.6% of high scored evaluations. On the other hand, scores of 1 or 2 were infrequent. For instance, only 2% of answers were scored 1.

With regards to the quality of sentences, results are more homogeneous. 58% of sentences scored between 2 and 4, with very little discrepancy. The number of sentences scored 1 is greater (7%) than in the evaluation on data in the sentence. The C+IE system showed the best results once again, although in this case, only 42% of answers scored 5.

Observing these evaluations, it seems that evaluators expressed that sentences are mostly good in transmitting data on hotels, despite their apparent lack of quality. Nevertheless, as expressed above, we detect some discrepancies between evaluations of the same sentences, which suggests it may be necessary to teach evaluators how to correctly assess sentences.

5.4.4 Sentences

Worst Sentences

Tables 5.11 and 5.12 show some of the worst classified sentences. Respectively, they present the worst sentences in terms of expressing the data they should express, and the lowest quality sentences, regarding grammaticality and semantics. By ‘worst’, we mean sentences that have had an average score below 2. We had only two examples for the data in the sentence evaluation and seven for sentence quality. Unsurprisingly, some of the sentences in first table are also in the second table.

Sentences are produced from a set of data, provided from the database. When we

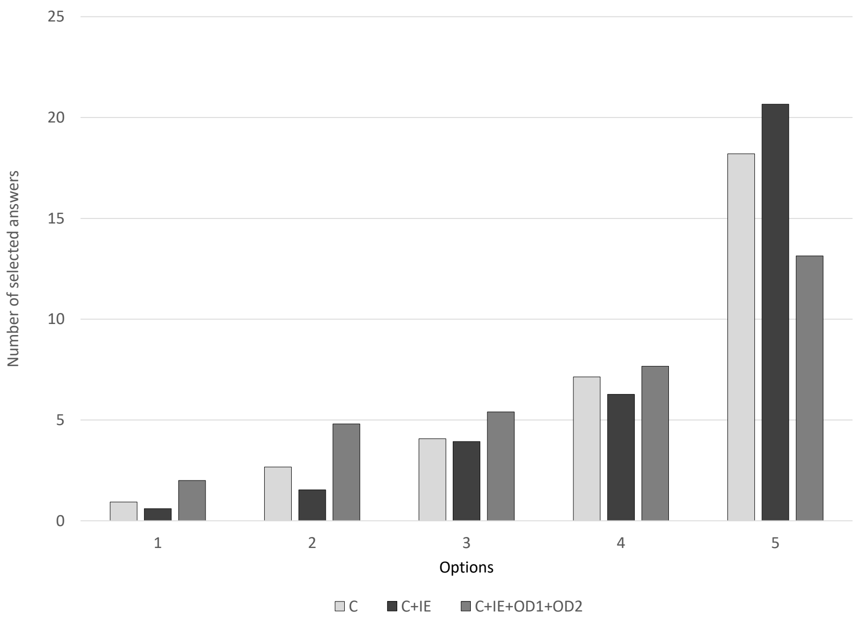


Figure 5.14: Number of selected answers, by option. Evaluation of the data expressed in sentences.

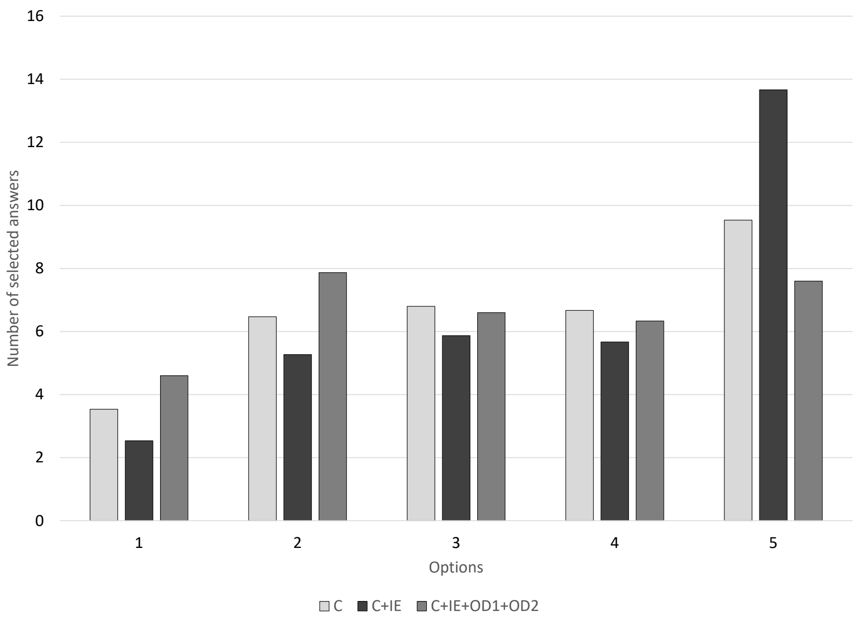


Figure 5.15: Number of selected answers, by option. Evaluation of sentence quality.

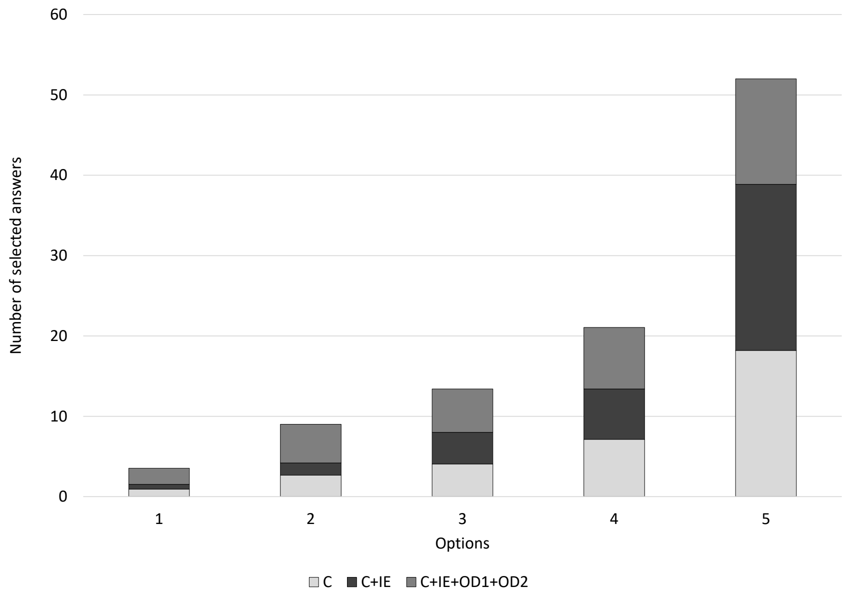


Figure 5.16: Relation between systems and their evaluation – the case of data expressed in sentences.

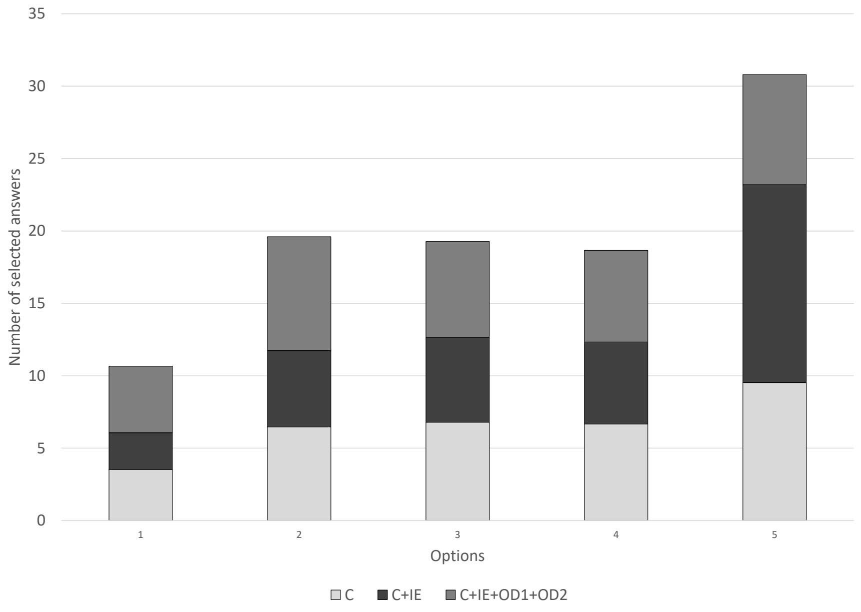


Figure 5.17: Relation between systems and their evaluation – the case of sentence quality.

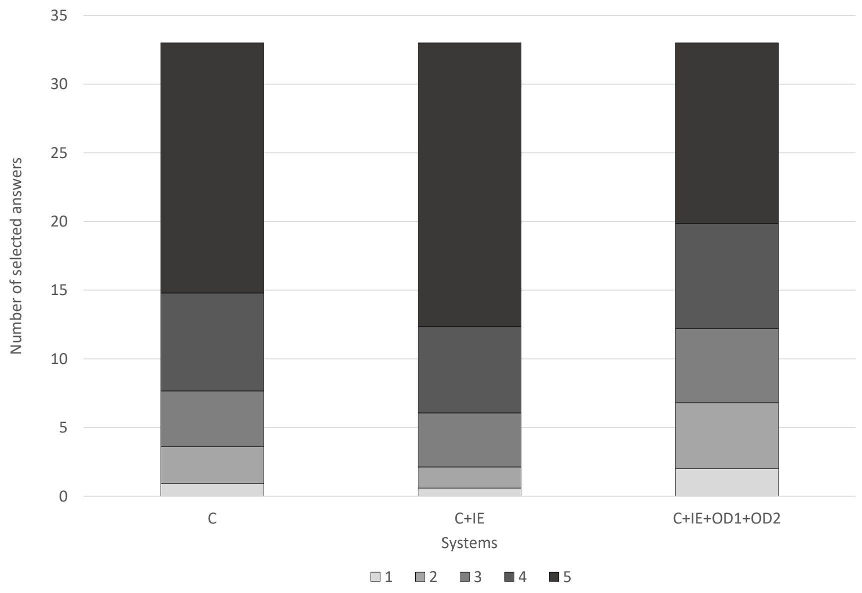


Figure 5.18: Number of evaluations by scores – the case of data expressed in sentences.

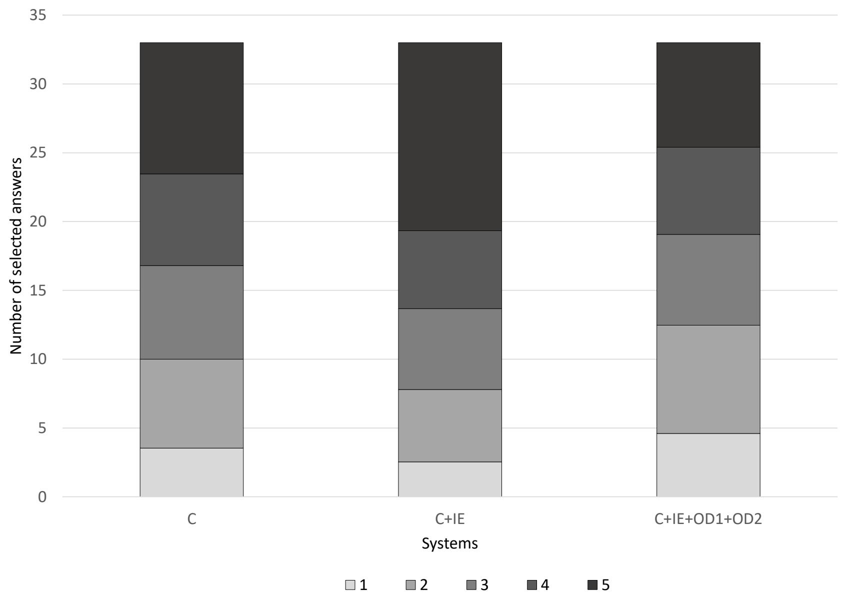


Figure 5.19: Number of evaluations by scores – the case of sentence quality.

analyze this data, it is quite amazing how the sentences were produced. For instance, in the sentences ID 78 and 42 almost no data was used, and the sentences reflect that lack of data.

Table 5.11: Worst sentences in the evaluation – data expressed in the sentence.

Sentence ID	Variant	Average Value	Standard Deviation	Sentence
78	C+IE+OD1+OD2	1.89	1.15	o hotel, algumas pessoas
80	C+IE+OD1+OD2	1.93	0.57	no 'Há Mar ao Luar' o serviço impessoal e pessoas

Table 5.12: Worst sentences in the evaluation – sentence quality.

Sentence ID	Variant	Average Value	Standard Deviation	Sentence
78	C+IE+OD1+OD2	1.47	0.99	o hotel, algumas pessoas
2	C	1.53	0.64	a limpeza do 'NorthSpot', por muitas pessoas
80	C+IE+OD1+OD2	1.53	0.64	no 'Há Mar ao Luar' o serviço impessoal e pessoas
8	C	1.80	0.68	o quarto do 'CR - Correeiros', por poucas pessoas, como péssimo
42	C+IE	1.87	0.99	o pessoal do hotel para muitas pessoas
79	C+IE+OD1+OD2	1.93	0.80	no 'Casa Spa d'Alma' é boa e de serviço
90	C+IE+OD1+OD2	1.93	0.88	a banheira é escorregadia e pessoas

Best Sentences

As we said before, the majority of our sentences have good quality. The following tables present some of best sentences evaluated. Table 5.13 shows sentences that scored 5 (or nearly 5) for the data expressed by the sentence. Table 5.14 shows good quality sentences which achieved the highest score for sentence quality. Comparing the sentences with higher scores to the data supplied for them, from the database, we can conclude that higher evaluations are closely linked to good data provision.

Table 5.13: Best sentences in evaluation regarding data expressed in the sentence.

Sentence ID	Variant	Average Value	Standard Deviation	Sentence
22	C	5.00	0	na 'Cayres Suites Centromar' a cama é confortável e macia
31	C	5.00	0	a banheira do hotel estava danificada ou não existia
40	C+IE	5.00	0	o estacionamento é bom
43	C+IE	5.00	0	o buffet era mau
35	C+IE	4.93	0.25	o buffet é bom para a esmagadora maioria das pessoas

Table 5.14: Best sentences in evaluation regarding sentence quality.

Sentence ID	Variant	Average Value	std	Sentence
35	C+IE	5.00	0	o buffet é bom para a esmagadora maioria das pessoas
40	C+IE	5.00	0	o estacionamento é bom
94	C+IE+OD1+OD2	5.00	0	o mobiliário era bonito
31	C	4.93	0.26	a banheira do hotel estava danificada ou não existia
49	C+IE	4.93	0.26	cerca de metade das pessoas referiram que o hotel é vergonhoso

5.5 Conclusions

In this chapter a new system was presented, based on a new scenario which was more challenging than the first one. Based on the results described in Chapter 3, only the *phrase-based* version was made. New corpora were defined, collected, expanded and processed, adding up to approximately 3 weeks of work.

Using the BLEU and Meteor automatic metrics, the corpora were evaluated and the results compared to the previous ones. Results show a slight decrease in performance, from our first system (Medication2PT), caused by the major difficulties of Summaries2Hotel Managers. Since an expanded corpus was used to build the language model, an evaluation of its effect was carried out as well. Surprisingly, the systems in which the language model was trained with only the output corpus obtained slight better results than the systems trained with the expanded corpus. When using an expanded corpus, results show that the use of an out-of-domain corpus in the training of the language model, provides better results than exclusively using an in-domain corpus.

A human evaluation was performed on a set of randomly chosen generated sentences. Results show that sentences generally obtained a good score. Only about 10% of sentences were rejected by evaluators. An analysis of those sentences led us to the conclusion that a possible reason for that lack of quality was the small amount of data used in their production. Results suggest that evaluators showed small discrepancies in evaluating the 'data' expressed in sentences, whereas on evaluation of sentence quality they are usually in agreement.

Chapter 6

Conclusions and Future Directions

In this chapter we conclude the thesis by summarizing the main tasks performed, going over the main results and contributions and suggesting relevant directions for the work to be taken further.

6.1 Work Summary

The increasing use of digital devices, such as computers, smartphones or tablets, or even what is known as the Internet-of-Things, allow for the collection of huge amounts of data. After being processed, this data should be returned to human users. The proliferation of these kinds of new devices brings with it new users and new situations in which devices are used, which in turn leads to the appearance of new demands.

This work was motivated by the need to provide this output to users in natural language, as well as the recent advances in multimodal interaction. These two aspects, led us to the need to find solutions for transforming data held by systems into a format which is natural to humans, such as written text, image output, the human voice or other kind of sounds.

From the outset, due to the context where it began, Portuguese was the language selected for the development of the work. Portuguese is one of the most used language in the world, and there are very few tools available for NLG. The first major effort addressed the exploration of the existing state-of-the-art systems, resources and tools that could be of use in the development of natural language generation in Portuguese. From this effort, which included experimentation with many publicly accessible tools, a clear picture emerged – no readily available tools meet our initial needs for Portuguese. The few tools found are generally developed in Brazilian Portuguese, and are focused on solving a specific issue.

At this stage, we came into contact with some state-of-the art systems in data-to-text. Systems like BabyTalk, SimpleNLG, NaturalOWL or Mountain were studied. All of those systems used different approaches. Some use dedicated computer programs, written with JAVA or C. Some used XML and ontologies. Others use a statistical approach to make the generation of natural language possible.

Our main goal was to develop a natural language generation system with very scarce resources and a very limited linguistic knowledge. Nevertheless, we were trying to find a way to produce a system which could produce a variety of quality utterances. Out of all the tools analyzed, the approach taken by Mountain interested us the most. Mountain uses the translation engine Moses as the main engine to generate new utterances in a given language. Despite the difficulty in setting up a Moses system, it is starting to be widely used, in different scenarios (Dehdari, 2015, Koehn, 2016). This convinced us that by adopting Mountain's approach, using the translation based data-driven methods, would be the best option to achieve our aims.

To gain knowledge and a proof-of-concept, we first decided to create a simple module capable of producing sentences for a Multimodal Medication Assistant in development, at the time, by the IEETA group headed by António Teixeira, in the scope of the *Smartphones for Seniors* QREN project. A statistical machine translation system uses an aligned bilingual text corpora. With that in mind, the input format for the natural language generation module was defined and a small corpus was defined and collected. Using the same technique as Mountain, the collected corpus was expanded. With that corpus, several variants were produced and trained, exploring techniques such as phrase-based and syntax-based generation. The resulting module was integrated for the first time in the smartphone-based Medication Assistant application.

The evaluation of our system was done with BLEU and Meteor scores. As we realized that these scores penalized the utterances produced by our system, we decided to carry out a human evaluation of the system's output as well. The idea was to confirm the evaluation made by the automatic evaluators.

Our translation based generation module produced sentences of variable quality. Therefore, we decided to investigate how to complement it. The challenge was to produce quality utterances in a simpler way, to replace sentences with a low evaluation. Our choice was to build a template-based component.

With that in mind, the first task was to develop a module that would evaluate the quality of the utterances generated. This module would classify and count the words in a sentence, then analyze the results and determine if the sentence should be marked as acceptable or not. If a sentence was marked as unacceptable, a second module would be used. Its function is to use the same data that was used to produce the rejected utterance, and produce a new utterance. This new utterance is generated by a templates module. This was chosen in order to ensure the correctness of the new utterance, albeit at the expense of variability and richness. The templates module was made with XML and XLST. These technologies were chosen because they are freely available as well as easy to use and integrate with other modules.

As a result of adding these 2 modules, our system has evolved into a hybrid proposal. This new version – the Hybrid NLG system – still conforms to our initial proposal, and main objective.

With the aim of proving their potential for use in more complex scenarios, our final task was to create a new system. The scenario chosen is the field of hotel evaluation. This development of this system was combined with ongoing work at IEETA, in the field of Information Extraction to create a web-to-humans (W2H) system. The aim is to provide small written text summaries of the most relevant information extracted from comments on hotels and their services. The focus of the work was only the first module of our approach – the translation system. After the definition of the data available, the tokens needed for the input language were defined, and a corpus was collected and expanded. Then, the resulting sentences were evaluated by automatic metrics and by human evaluators.

6.2 Main Results

The main outcome of this thesis is the demonstration that it is possible to create systems capable of translating information/data into decent-quality sentences in Portuguese. This is done without a major effort in terms of resources creation and with the common knowledge of an experienced application developer. The systems created, particularly the hybrid system, are capable of providing a good solution for problems in data to text conversion.

The second major result of this work is the process of creating a solution for data-to-text with scarce resources and limited know-how. A First Hybrid translation+template based NLG system for specific domain was proposed and published.

Other relevant results were:

- The use of Translation based NLG systems with the Portuguese language. To the best of our knowledge, this is the first time that this has been done with the language of Camões;

- The combination of Statistical Machine Translation tools and templates, which helps demystify the use of the latter. Effectively, current publications portray Templates as an out-of-fashion and obsolete technology. We proved that this may not be entirely true, and that templates are still a good solution as a complement to more complex systems;

- The use in scenarios which are normally out of the scope of translation based approaches, specifically, the evaluation of Hotels. Our system is still in early stages of development, but is clearly aiming higher than others that have been published;

- The production of new corpora. We proved that even with scarce resources, is possible to build and expand corpora with acceptable results;

- The integration into applications: first, with Medication Assistant; and lastly, with an Information Extraction application.

6.3 Future Work

Our systems and modules are interesting solutions for data-to-text problems in the scope of human-machine multimodal interaction, but they are not perfect. Several paths can be taken in order to continue developing each of the modules and systems. Some ideas are put forward over the next subsections:

6.3.1 Improving the Modules

Translation Module

Some of the sentences produced by our systems were low in quality. One reason for this is the poor data supplied to the translation module. Another reason is certainly the relatively small number of sentences that form our corpora. We are aware of it, but what size should the corpus be in order to produce good utterances? That is one question that we will pursue in the quest to improve our systems.

Moses is the engine used to process the statistical translation from input language to output language. Nothing about our work forces us to use this tool. Nevertheless, Moses is the most used engine in this field and a new version[1] was recently released. It promises to be faster than previous one and have more interesting features. We intend to study this new release, and find out how to improve the accuracy of the utterances produced by manipulating the several parameters it features.

Quality Evaluation Module

One of our biggest challenges was finding tools that process sentences in Portuguese, as there are very few with acceptable quality levels. Two of them are the parser and the features extraction, which are used to evaluate the utterances produced by Moses, classifying each word of a sentence. Finding a better tool is another goal we would like to achieve.

In future work, we intend to increase the number of features involved in our system, and particularly, increase the accuracy of those which evaluate the grammar and syntax of sentences. That would allow us to evaluate utterances better, reducing the false positives (sentences that are with poor quality, but classified as good) and the false negatives (sentences that have good quality, but classified as bad). The improvements mentioned here will result in a better system, which will resort less to the template module.

Aggregation Module

There is one further task we would like to achieve and implement – the aggregation of related sentences. This module will be responsible for identifying the noun phrase (subject) and the verb phrase (the feature evaluated) of sentences. Then, it will suppress the subject of second sentence and join both sentences, producing a compound sentence. The use of coordinating conjunctions (and - but - or - nor - so - then - yet) will depend on the agreement of the original sentences. This module will be designed for a second system, but nothing stops it from being used with our first system, with the necessary adaptations.

[1]http://www.statmt.org/moses/?n=Site.Moses2

6.3.2 Integration of the new and improved NLG prototype systems in Applications

The full integration of both systems in real applications is an ongoing process. We carried out a preliminary integration of our first system – Medication Assistant – in the S4S project. We expect to process a full integration in the near future.

We expect to complete our second system – Summaries of the Evaluation of Hotel Services by their customers – within a short time, with the definition and production of the templates module. This work is integrated in an MSc project, at IEETA. At the end, we expect to integrate it in the final release of a system aimed at collecting and extracting opinions from hotel costumers. After processing the data, the system should be capable of presenting a short text that sums up the opinions collected.

6.3.3 New Fields

Last, but not least, for us it is important to study and implement ways of rapidly extending this work to new applications. The use of multimodal interfaces is a major interest too. Only through the use of speech and/or images will it be possible to complement the utterances delivered, so as to reduce or even eliminate errors in communication.

Bibliography

Agência Lusa (2010). Falantes de português irão aumentar para 335 milhões em 2050. Online, https://www.publico.pt/culturaipsilon/noticia/falantes-de-portugues-irao-aumentar-para-335-milhoes-em-2050-1429372. Visited on November 05, 2016.

Aha, D. W., Kibler, D., and Alvert, M. K. (1991). Instance-based learning algorithms. *Machine Learning*, 6:37–66.

Alemzadeh, H. and Devarakonda, M. (2017). An NLP-based cognitive system for disease status identification in electronic health records. In *2017 IEEE EMBS International Conference on Biomedical Health Informatics (BHI 2017)*, pages 89–92, Orlando, Florida, USA.

Amado, R., Matias, C., and Felgueiras, C. (2016). Aumenta número de falantes de Língua Portuguesa. Online, http://www.rtp.pt/noticias/pais/aumenta-numero-de-falantes-de-lingua-portuguesa_v962257. Visited on November 06, 2016.

Araújo, R., Oliveira, R., Novais, E., Tadeu, T., Pereira, D., and Paraboni, I. (2010). SINotas: the Evaluation of a NLG Application. In *Seventh International Conference on Language Resources and Evaluation (LREC)*, pages 2388–2391, Valletta, Malta.

Aulia, I. and Barmawi, A. M. (2015). An automatic health surveillance chart interpretation system based on indonesian language. In *2015 International Conference on Advanced Computer Science and Information Systems (ICACSIS2015)*, pages 163–170, Depok, West Java, Indonesia.

Bateman, J. and Zock, M. (2004). Natural language generation. In Mitkov, R., editor, *The Oxford Handbook of Computational Linguistics*, pages 284–304. Oxford University Press.

Bates, M. (1995). Models of natural language understanding. In *Proceedings of the National Academy of Sciences of the United States of America*, volume 92 (22), pages 9977–9982, Irvine, California, USA.

Belz, A., White, M., Espinosa, D., Kow, E., Hogan, D., and Stent, A. (2011). The first surface realisation shared task: Overview and evaluation results. In *Proceedings of the 13th European Workshop on Natural Language Generation (ENLG '11)*, ENLG '11, pages 217–226, Stroudsburg, PA, USA. Association for Computational Linguistics.

Bernsen, N. O. (2007). Multimodality Theory. In Tzovaras, D., editor, *Multimodal User Interfaces – From Signals to Interaction*, Signals and Commmunication Technologies, chapter 2, pages 5–29. Springer Berlin Heidelberg.

Bollmann, M. (2011). Adapting SimpleNLG to German. In *Proceedings of the 13th European Workshop on Natural Language Generation (ENLG 2011)*, pages 133–138, Nancy, France. Association for Computational Linguistics.

Bolt, R. A. (1980). "Put-that-there": Voice and gesture at the graphics interface. In *7th Annual Conference on Computer Graphics and Interactive Techniques*, Seattle, Washington, United States. ACM.

Branco, A. and Silva, J. a. (2004). Evaluating Solutions for the Rapid Development of Stat-of-the-Art POS Taggers for Portuguese. In Lino, M. T., Xavier, M. F., Ferreira, F., Costa, R., and Silva, R., editors, *4th International Conference on Language Resources and Evaluation (LREC2004)*, pages 507–510, Paris, France.

Breiman, L. (2001). Random Forests. *European Journal of Mathematics*, 45:5–32.

Busemann, S. (1996). Best-First Surface Realization. In *Proceedings of 8th International Workshop on Natural Language Generation (INLG-96)*, pages 101–110, Herstmonceux, Sussex, UK.

Busemann, S. (2005). Best-First Surface Realization. In Wilcock, G., Jokinen, K., Mellish, C., and Reiter, E., editors, *Proceedings of the 10th European Workshop on Natural Language Generation (ENLG-05)*, pages 32–39, Aberdeen, Scotland, UK.

Busemann, S. and Horacek, H. (1998). A Flexible Shallow Approach to Text Generation. In *Proceedings of the 9th International Workshop on Natural Language Generation*, pages 238–247, Niagara-on-the-Lake, Canada. Association for Computational Linguistics.

Channarukul, S. (1999). Yag: A template-based natural language generator for real-time systems. Technical report, Department of Electrical Engineering and Computer Science, University of Wisconsin-Milwaukee, USA.

Chawla, N. V., Bowyer, K. W., Hall, L. O., and Kegelmeyer, W. P. (2002). SMOTE: Synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research*, 16:321–357.

Chiarcos, C., Lang, M., and Verhagen, P. (2015). IT-assisted Exploration of Excavation Reports. Using Natural Language Processing in the Archaeological Research Process. In *Proceedings of the 43rd Annual Conference on Computer Applications and Quantitative Methods in Archaeology (CAA2015)*, pages 87–94, University of Siena, Italy. Archaeopress Publishing Ltd.

Coetzee, L., Viviers, I., and Barnard, E. (2009). Model based estimation for multimodal user interface component selection. In *20th Annual Symposium of the Pattern Recognition Association of South Africa (PRASA 2009)*, pages 1–6, South Africa. PRASA 2009.

Deemter, K. V., Krahmer, E., and Theune, M. (1999). Plan-based vs. template-based NLG: a false opposition? In *Proceedings of the KI Workshop 'May I speak freely?: Between templates and free choice in NLG'*, Saarbrücken, Germany. DFKI.

Deemter, K. V., Krahmer, E., and Theune, M. (2005). Real versus Template-Based Natural Language Generation: A False Opposition? *Computational Linguistics*, 31(1):15–24.

Dehdari, J. (2015). Seminar on the Moses Machine Translation System. Online, `http://jon.dehdari.org/teaching/uds/moses/`. Visited on November 07, 2016.

Denkowski, M. and Lavie, A. (2011). Meteor 1.3 : Automatic Metric for Reliable Optimization and Evaluation of Machine Translation Systems. In *Proceedings of the 6th Workshop on Statistical Machine Translation - EMNLP 2011*, pages 85–91, Edinburgh, Scotland, UK.

Denkowski, M. and Lavie, A. (2014). Meteor Universal: Language Specific Translation Evaluation for Any Target Language. In *Proceedings of the Ninth Workshop on Statistical Machine Translation - EACL 2014*, pages 376–380, Baltimore, Maryland, USA. Association for Computational Linguistics.

Dumas, B., Lalanne, D., and Oviatt, S. (2009). Multimodal Interfaces: A Survey of Principles, Models and Frameworks. In Lalanne, D. and Kohlas, J., editors, *Human Machine Interaction*, volume 5540 of *LCNS*, pages 3–26. Springer-Verlag, Berlin/Heidelberg.

Dymetman, M., Lux, V., and Ranta, A. (2000). XML and Multilingual Document Authoring: Convergent Trends. In *Proceedings of the 18th Conference on Computational Linguistics - Volume 1*, COLING '00, pages 243–249, Stroudsburg, PA, USA. Association for Computational Linguistics.

Erk, K., Kowalski, A., Padó, S., and Pinkal, M. (2003). Towards a Resource for Lexical Semantics: A Large German Corpus with Extensive Semantic Annotation. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics - Volume 1 (ACL '03)*, ACL '03, pages 537–544, Stroudsburg, PA, USA. Association for Computational Linguistics.

Erk, K. and Padó, S. (2004). A powerful and versatile XML format for representing role-semantic annotation. In *In Proceedings of 4th International Conference on Language Resources and Evaluation (LREC 2004)*, pages 799–802, Lisbon, Portugal.

Fawcett, T. (2006). An Introduction to ROC Analysis. *Pattern Recogn. Lett.*, 27(8):861–874.

Felice, M. (2012). *Linguistic Indicators for Quality Estimation of Machine Translations.* Erasmus mundus international masters/ma in natural language processing & human language technology, Universitat Autònoma de Barcelona.

Ferreira, F., Almeida, N., Rosa, A. F., Oliveira, A., Casimiro Pereira, J., Silva, S., and Teixeira, A. (2014). Elderly Centered Design for Interaction – The Case of the S4S Medication Assistant. *Procedia Computer Science*, 27(Dsai 2013):398–408.

Ferreira, F., Almeida, N., Rosa, A. F., Oliveira, A., Teixeira, A., and Pereira, J. C. (2013). Multimodal And Adaptable Medication Assistant for the Elderly – A prototype for Interaction and Usability in Smartphones. In *Proceedings of 8th Iberian Conference on Information Systems and Technologies (CISTI 2013)*, pages 309–314, Lisbon, Portugal.

Fonseca, A. C. (1993). Comunicação em Linguagem Natural para um Tutor Inteligente. Master's thesis, Universidade Técnica de Lisboa, Instituto Superior Técnico Lisboa.

Foster, M. E. and White, M. (2004). Techniques for Text Planning with XSLT. In *Proceeedings of the Workshop on NLP and XML (NLPXML-2004): RDF/RDFS* *and OWL in Language Technology*, NLPXML '04, pages 1–8, Stroudsburg, PA, USA. Association for Computational Linguistics.

Gatt, A. and Reiter, E. (2009). SimpleNLG: a realisation engine for practical applications. In *Proceedings of the 12th European Workshop on Natural Language Generation*, pages 90–93, Athens, Greece. Association for Computational Linguistics.

Glahn, H. R. (1970a). Computer-produced worded forecasts. *Bulletin American Meteorological Society*, 51(12):1126–1131.

Glahn, J. R. (1970b). IFPS. Online, `http://www.nlg-wiki.org/systems/IFPS`. Visited on May 09, 2016.

Goldman, N. M. (1975). Sentence Paraphrasing from a Conceptual Base. *Communications of the ACM*, 18(2):96–106.

Grover, C., Klein, E., Lapata, M., and Lascarides, A. (2002). Xml-based nlp tools for analysing and annotating medical language. In *Proceedings of the 2Nd Workshop on NLP and XML - Volume 17*, NLPXML '02, pages 29–36, Stroudsburg, PA, USA. Association for Computational Linguistics.

Haarmann, B. and Sikorski, L. (2015). Natural Language News Generation from Big Data. *International Journal of Computer, Electrical, Automation, Control and Information Engineering*, 9(6):1496 – 1502.

Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., and Witten, I. H. (2009). The WEKA data mining software. *ACM SIGKDD Explorations Newsletter*, 11(1):10–18.

Hall, M., Witten, I., and Frank, E. (2011). *Data Mining: Practical Machine Learning Tools and Techniques.* Morgan Kaufmann, 3th edition.

Hastie, H. and Belz, A. (2014). A comparative evaluation methodology for nlg in interactive systems. In *9th International Conference on Language Resources and Evaluation*, pages 4004–4011.

Heimonen, T., Hakulinen, J., Sharma, S., Turunen, M., Lehtikunnas, L., and Paunonen, H. (2016). Multimodal Interaction in Process Control Rooms: Are We There Yet? In *Proceedings of 5th International Symposium on Pervasive Displays (PerDis'16)*, pages 20–32, Oulu, Finland. ACM.

Hunter, J., Freer, Y., Gatt, A., Reiter, E., Sripada, S., Sykes, C., and Westwater, D. (2011). BT-Nurse: computer generation of natural language shift summaries from complex heterogeneous medical data. *Journal of the American Medical Informatics Association*, 18(5):621–624.

Hunter, J., Reiter, E., Sripada, Y. G. S., and Yu, J. (2005). SumTime. Online, `https://www.abdn.ac.uk/ncs/departments/computing-science/sumtime-317.php`. Visited on November 05, 2014.

IRST (2011). IRST Language Modeling Toolkit. Online, `http://hlt-mt.fbk.eu/technologies/irstlm`. Visited on November 07, 2014.

Jokinen, K. and Wilcock, G. (2003). Adaptivity and response generation in a spoken dialogue system. In van Kuppevelt, J. and Smith, R. W., editors, *Current and New Directions in Discourse and Dialogue*, pages 213–234. Springer Netherlands, Dordrecht.

Jurafsky, D. and Martin, J. H. (1999). *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics and Speech Recognition, 1st Edition.* Pearson - Prentice Hall, New Jersey - USA, 1st. edition.

Karpov, A., Carbini, S., Ronzhin, A., and Viallet, J. E. (2008). Two SIMILAR Different Speech and Gestures Multimodal Interfaces. In Tzovaras, D., editor, *Multimodal User Interfaces: From Signals to Interaction*, pages 155–184. Springer Berlin Heidelberg, Berlin, Heidelberg.

Koehn, P. (2004). Pharaoh: A beam search decoder for phrase-based statistical machine translation models. In Frederking, R. and Taylor, K., editors, *Machine Translation: From Real Users to Research*, volume 3265 of *Lecture Notes in Computer Science*, pages 115–124. Springer -Verlag Berlin Heidelberg.

Koehn, P. (2014). MOSES: Statistical Machine Translation System - User Manual and Code Guide. Online, http://www.statmt.org/moses/manual/manual.pdf. Visited on November 06, 2014.

Koehn, P. (2016). Moses: bringing machine translation to the masses. Online, `http://www.ed.ac.uk/research/impact/science-engineering/moses`. Visited on November 07, 2016.

Koehn, P., Hoang, H., Birch, A., Callison-Burch, C., Federico, M., Shen, W., Moran, C., Zens, R., Bojar, O., Constantin, A., and Herbst, E. (2007). Moses: Open source toolkit for statistical machine translation. In *45th Annual Meeting of the*

Association for Computational Linguistics Companion Volume Proceedings of the Demo and Poster Sessions, pages 177–180, Prague, Czech Republic. Association for Computational Linguistics.

Kohavi, R. (1995). A Study of Cross-validation and Bootstrap for Accuracy Estimation and Model Selection. In *Proceedings of the 14th International Joint Conference on Artificial Intelligence - Volume 2*, IJCAI'95, pages 1137–1143, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Konstantopoulos, S., Androutsopoulos, I., Baltzakis, H., Karkaletsis, V., Matheson, C., Tegos, A., and Trahanias, P. (2008). Indigo: Interaction with personality and dialogue enabled robots [system demonstration]. In *18th European Conference on Artificial Intelligence (ECAI)*, Patras, Greece.

Langner, B. (2010). *Data-driven Natural Language Generation: Making Machines Talk Like Humans Using Natural Corpora*. Phd, School of Computer Science – Carnegie Mellon University, Pittsburgh, PA 15213, USA.

Langner, B. and Black, A. W. (2009). MOUNTAIN: A Translation-based Approach to Natural Language Generation for Dialog Systems. In *First International Workshop on Spoken Dialogue Systems Techology (IWSDS 2009)*, pages 1109–1112, Irsee, Germany.

Langner, B., Vogel, S., and Black, A. W. (2010). Evaluating a Dialog Language Generation System: Comparing the Mountain System to Other NLG Approaches . In *INTERSPEECH 2010 - 11th Annual Conference of the International Speech Communication Association*, Makuhari, Chiba, Japan.

Law, A. S., Freer, Y., Hunter, J., Logie, R. H., McIntosh, N., and Quinn, J. (2005). A comparison of graphical and textual presentations of time series data to support medical decision making in the neonatal intensive care unit. *Journal of Clinical Monitoring and Computing*, 19(3):183–194.

Legin, A., Rudnitskaya, A., Seleznev, B., and Vlasov, Y. (2005). Electronic tongue for quality assessment of ethanol, vodka and eau-de-vie. *Journal of Analytica Chimica Acta – Interpretation, Design and Selection of Biomolecular Interactions*, 534(1):129–135.

Lemon, O. (2011). Learning what to say and how to say it: Joint optimisation of spoken dialogue management and natural language generation. *Computer Speech & Language*, 25(2):210–221.

Machado, M. J., Fontes, H. L., and Rosas, J. a. (2010). Sistema de tradução automática Moses e Moses for Mere Mortals. *A Folha*, pages 22–25.

Mairesse, F., Gašić, M., Jurčíček, F., Keizer, S., Thomson, B., Yu, K., and Young, S. (2010). Phrase-based Statistical Language Generation Using Graphical Models and Active Learning. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, ACL '10, pages 1552–1561, Stroudsburg, PA, USA. Association for Computational Linguistics.

Mairesse, F. and Walker, M. (2007). PERSONAGE: Personality Generation for Dialogue. In *In Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics (ACL-2007)*, pages 496–503, Prague, Czech Republic.

Mairesse, F. and Walker, M. A. (2010). Towards personality-based user adaptation: psychologically informed stylistic language generation. *User Modeling and User-Adapted Interaction*, 20(3):227–278.

Mairesse, F. and Walker, M. A. (2011). Controlling User Perceptions of Linguistic Style: Trainable Generation of Personality Traits. *Computational Linguistics*, 37(3):455–488.

Mairesse, F. and Young, S. (2014). Stochastic Language Generation in Dialogue Using Factored Language Models. *Journal Computational Linguistics*, 40(4):763–799.

Mandelbrot, B. B. (1989). Fractals and an art for the sake of science. In *SIGGRAPH 89 Art Show Catalog - Computer Art in Context*, SIGGRAPH '89, pages 14–21, New York, NY, USA. ACM.

Manning, C. D., Raghavan, P., and Schütze, H. (2008). *Introduction to Information Retrieval.* Cambridge University Press, Cambridge, England.

Mazzei, A., Battaglino, C., and Bosco, C. (2016). SimpleNLG-IT: adapting SimpleNLG to Italian. In *Proceedings of the 9th International Natural Language Generation conference (INLG2016)*, pages 184–192, Edinburgh, UK. Association for Computational Linguistics.

McCauley, L., D'Mello, S., Kim, L., and Polkosky, M. (2008). MIKI: A Case Study of an Intelligent Kiosk Avatar and its Usability. In Magnenat-Thalmann, N., Jain, L. C., and Ichalkaranje, N., editors, *New Advances in Virtual Humans*, Studies in Computational Intelligence, Vol. 140, VIII, pages 153–176. Springer-Verlag, Berlin/Heidelberg.

McRoy, S. W., Channarukul, S., and Ali, S. A. (2000). YAG: A Template-Based Generator For Real-Time Systems. In *Proceedings of the First International Natural Language Generation Conference (INLG'2000)*, pages 264–267, Mitzpe Ramon, Israel.

Mellish, C., Scott, D., Chaill, L., Paiva, D., Evans, R., and Reape, M. (2006). A Reference Architecture for Natural Language Generation Systems. *Journal of Natural Language Engineering*, 12(1):1–34.

Mendes, M. D. (2004). Relações lexicais na geração de língua natural. Master's thesis, Universidade de Coimbra - Portugal.

Mitchell, M., Bohus, D., and Kamar, E. (2014). Crowdsourcing language generation templates for dialogue systems. In *Proceedings of the INLG and SIGDIAL 2014 Joint Session*, pages 16–24, Philadelphia, Pennsylvania, U.S.A. Association for Computational Linguistics.

Moore, J., Foster, M. E., Lemon, O., and White, M. (2004). Generating tailored, comparative descriptions in spoken dialogue. In *Proceedings of FLAIRS 2004*. American Association for Artificial Intelligence.

Moses (2011). Moses – statistical machine translation system. Online, `http://www.statmt.org/moses/`. Visited on June 27, 2011.

Moses (2014). Moses – Baseline System. Online, `http://www.statmt.org/moses/?n=Moses.Baseline`. Visited on November 11, 2014.

Nallaperumal, K. and Krishnan, A. (2015). *Engineering Research Methodology*. PHI Learning Private Limited, New Delhi, India, 1st edition.

Narayan, K., Isbell, C., and Roberts, D. (2011). DEXTOR: Reduced Effort Authoring for Template-Based Natural Language Generation. In *Proceedings of the Seventh AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, Palo Alto, California, USA. Association for the Advancement of Artificial Intelligence (www.aaai.org).

Nass, C., Fogg, B., and Moon, Y. (1996). Can computers be teammates? *International Journal of Human-Computer Studies*, 45(6):669 – 678.

National Academy of Engineering (1995). *Forces Shaping the U.S. Academic Engineering Research Enterprise*. The National Academies Press, Washington, DC, USA.

Noor, A. K. and Aras, R. (2015). Potential of multimodal and multiuser interaction with virtual holography. *Journal of Advances in Engineering Software*, 81(1):1–6.

Novais, E. M., Oliveira, R. L., Pereira, D. B., Tadeu, T. D., and Paraboni, I. (2009). A Testbed for Portuguese Natural Language Generation. In *2009 Seventh Brazilian Symposium in Information and Human Language Technology*, pages 154–157, São Carlos, São Paulo, Brazil. IEEE.

Och, F. J. (2011). GIZA++ statistical translation models toolkit. Online, `https://github.com/moses-smt/giza-pp`. Visited on July 02, 2011.

Oliveira, H. G. (2012). PoeTryMe: a versatile platform for poetry generation. In *Proceedings of the ECAI 2012 Workshop on Computational Creativity, Concept Invention, and General Intelligence*, C3GI 2012, Montpellier, France.

Oliveira, R. D. and Sripada, S. (2014). Adapting SimpleNLG for Brazilian Portuguese realisation. In *Proceedings of the 8th International Natural Language Generation Conference (INLG 2014)*, pages 93–94, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Os, E. D. and Boves, L. (2003). Towards ambient intelligence: Multimodal computers that understand our intentions. In *Proceedings of eChallenges e-2003*, pages 22–24, Palazzo Re Enzo, Bologna, Italy.

Oviatt, S. (1997). Multimodal interactive maps: designing for human performance. *Human-Compututer Interaction*, 12(1):93–129.

Oviatt, S. (2003). Advances in Robust Multimodal Interface Design. *IEEE Computer Graphics and Applications*, 23(5):62–68.

Oya, T., Mehdad, Y., Carenini, G., and Ng, R. (2014). A template-based abstractive meeting summarization: Leveraging summary and source text relationships. In *Proceedings of the 8th International Natural Language Generation Conference (INLG)*, pages 45–53, Philadelphia, Pennsylvania, U.S.A. Association for Computational Linguistics.

Padró, L. (2011). Analizadores multilingües en freeling. *Linguamatica*, 3(2):13–20.

Padró, L. and Stanilovsky, E. (2012). Freeling 3.0: Towards wider multilinguality. In *Proceedings of the Language Resources and Evaluation Conference (LREC 2012)*, Istanbul, Turkey. ELRA.

Papineni, K., Roukos, S., Ward, T., and Zhu, W.-J. (2002). BLEU: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania. Association for Computational Linguistics.

Pereira, J. C. and Teixeira, A. (2015). Geração de Linguagem Natural para Conversão de Dados em Texto - Aplicação a um Assistente de Medicação para o Português. *Linguamática*, 7(1):3–21.

Portet, F., Reiter, E., Gatt, A., Hunter, J., Sripada, S., Freer, Y., and Sykes, C. (2009). Automatic generation of textual summaries from neonatal intensive care data. *Artifitial Intelligence*, 173(7-8):789–816.

Ramos-Soto, A., Bugarín, A., and Barro, S. (2016). On the role of linguistic descriptions of data in the building of natural language generation systems. *Journal of Fuzzy Sets and Systems*, 285(1):31–51.

Ramos-Soto, A., Pereira-Fariña, M., Bugarín, A., and Barro, S. (2015). A model based on computational perceptions for the generation of linguistic descriptions of data. In *2015 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*, pages 1–8.

Reeves, B. and Nass, C. (1996). *The Media Equation: How People Treat Computers, Television, and New Media Like Real People and Places.* Cambridge University Press, California, USA, 1st edition.

Reiter, E. (1994). Has a Consensus NL Generation Architecture Appeared and is it Psycholinguistically Plausible? In *Proceedings of the Seventh International Natural Language Generation Workshop (INLG'94)*, page 163–170, Kennebunkport, Maine. Association for Computational Linguistics.

Reiter, E. (1995). Nlg vs. templates. In *Proceedings of the Fifth European Workshop on Natural-Language Generation (ENLGW-1995)*, ENLGW-1995, pages 95–105, Leiden, The Netherlands.

Reiter, E. (2007). An architecture for data-to-text systems. In *Proceedings of the Eleventh European Workshop on Natural Language Generation (ENLG 2007)*, pages 97–104, Germany. Association for Computational Linguistics.

Reiter, E. (2010). Natural Language Generation. In Clark, A., Fox, C., and Lappin, S., editors, *The Handbook of Computational Linguistics and Natural Language Processing*, chapter 20, pages 574–598. Wiley-Blackwell, USA.

Reiter, E. (2016). SimpleNLG - Java API for Natural Language Generation. Online, `https://github.com/simplenlg/simplenlg`. Visited on December 16, 2016.

Reiter, E. and Belz, A. (2009). An Investigation into the Validity of Some Metrics for Automatically Evaluating Natural Language Generation Systems. *Computational Linguistics*, 35(4):529–558.

Reiter, E. and Dale, R. (1997). Building applied natural language generation systems. *Journal of Natural Language Engineering - Cambridge University Press*, 3(1):57–87.

Reiter, E. and Dale, R. (2000). *Building Natural Language Generation Systems*. Cambridge University Press.

Ribeiro, A. (1995). Natural Language Generation with Rhetorical Relations and Focus Theory. Master's thesis, Edinburgh University – UK.

Rocha, P. and Santos, D. (2016). CETEMPúblico (Corpus de Extractos de Textos Electrónicos MCT/Público) . Online, `http://www.linguateca.pt/cetempublico/`. Visited on September 05, 2016.

Rodrigues, M. J. F. (2013). *Model of Access to Natural Language Sources in Electronic Government*. PhD thesis, Aveiro University - Portugal.

Rosas, J. a. L., Fontes, H. L., and Machado, M. J. (2014). Moses for Mere Mortals (MMM). Online, `https://github.com/jladcr/Moses-for-Mere-Mortals`. Visited on September 15, 2014.

Rosenthal, R. and Rosnow, R. L. (1991). *Essentials of Behavioral Research: Methods and Data Analysis*. McGraw-Hill Series in Psychology, New York, USA, 2nd edition.

Rousseau, C., Bellik, Y., Vernier, F., and Bazalgette, D. (2006). A framework for the intelligent multimodal presentation of information. *Journal of Signal Processing - Special section: Multimodal human-computer interfaces*, 86(12):3696–3713.

Ruth, D. P. and Peroutka, M. R. (1993). The interactive computer worded forecast. In *Preprints of the 9th International Conference on Interactive Information and Processing Systems for Meteorology, Oceanography, and Hydrology*, pages 321–326, Anaheim, California, USA. American Meteorological Society.

Salem, A. E., Dušan, T., and Nikola, M. (2014). Building ontologies for different natural languages. *Computer Science and Information Systems*, 11(2):623–644.

Salzberg, S. L. and Fayyad, U. (1997). On Comparing Classifiers: Pitfalls to Avoid and a Recommended Approach. *Data Mining and Knowledge Discovery*, pages 317–328.

Santos, D. (1992). Natural Language and Knowledge Representation. In *ERCIM 92*.

Santos, D. and Rocha, P. (2001). Evaluating CETEMPúblico, a free resource for Portuguese. In *Proceedings of the 39th Annual Meeting of the Association for Computational Linguistics (ACL '01)*, pages 442–449.

Schmid, H. (1995). Improvements in Part-of-Speech Tagging With an Application To German. In *Proceedings of the ACL SIGDAT Workshop*, Dublin, Ireland.

Schmid, H. (2015). TreeTagger – a language independent part-of-speech tagger. Online, `http://www.cis.uni-muenchen.de/~schmid/tools/TreeTagger/`. Visited on February 14, 2015.

Sebe, N. (2009). Multimodal interfaces: Challenges and perspectives. *J. Ambient Intell. Smart Environ.*, 1(1):23–30.

Silva Junior, D. F. P., Paraboni, I., and Novais, E. M. (2013). Um Sistema de Realização Superficial para Geração de Textos em Português. *RITA - Revista de Informática Teórica e Aplicada - Instituto de Informática da Universidade Federal do Rio Grande do Sul – Brasil*, 20(3):31–48.

Simões, A. (2016). Lingua-Jspell. Online, `http://search.cpan.org/~ambs/Lingua-Jspell/`. Visited on April 25, 2016.

Smedt, K. D., Horacek, H., and Zock, M. (1993). Architectures for Natural Language Generation: Problems and Perspectives. In Adorni, G. and Zock, M., editors, *Proceedings of Trends in Natural Language Generation, An Artificial Intelligence Perspective, Fourth European Workshop (EWNLG '93)*, pages 17–46, Pisa, Italy. Springer Berlin Heidelberg.

Soares, A. S. (2001). Gramática de Unificação Funcional: Levantamento de Requisitos para a Geração Sentencial de Português. Master's thesis, Instituto de Ciências Matemáticas e da Computação - USP/São Carlos - Brasil.

Sripada, S. G., Reiter, E., Hunter, J., and Yu, J. (2003). Exploiting a parallel TEXT-DATA corpus. In *Proceedings of the Corpus Linguistics 2003*, pages 734–743, Lancaster, U.K.

Stent, A. and Molina, M. (2009). Evaluating automatic extraction of rules for sentence plan construction. In *Proceedings of the SIGDIAL 2009 Conference: The 10th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 290–297, London, United Kingdom. Association for Computational Linguistics.

Stent, A., Prasad, R., and Walker, M. (2004). Trainable sentence planning for complex information presentation in spoken dialog systems. In *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*, page 79, Barcelona, Spain. Association for Computational Linguistics.

Stenzhorn, H. (2002). XtraGen: A Natural Language Generation System Using XML and Java technologies. In *Proceedings of the 2Nd Workshop on NLP and XML - Volume 17*, NLPXML '02, pages 59–66, Stroudsburg, PA, USA. Association for Computational Linguistics.

Teixeira, A., Almeida, N., Pereira, C., Silva, M. O. e., and Pereira, J. C. (2013a). Serviços de Suporte à Interação Multimodal. In Teixeira, A. J. S., Queirós, A., and da Rocha, N. P., editors, *Laboratório Vivo de Usabilidade - Living Usability Lab*, chapter 11, pages 151–165. ARC Publishing e Editores.

Teixeira, A., Ferreira, F., Almeida, N., Rosa, A. F., Pereira, J. C., Silva, S., Queirós, A., and Oliveira, A. (2013b). Multimodality and Adaptation for an Enhanced Mobile Medication Assistant for the Elderly. In *MOBACC 2013 - 3rd Workshop on Mobile Accessibility @ CHI 2013*, Paris, France.

Teixeira, A., Pereira, C., Silva, M. O. e., Pacheco, O., and Pereira, J. C. (2011a). AdaptO - Adaptive Multimodal Output. In *1st. International Conference on Pervasise and Embedded Computing and Communication Systems (PECCS 2011)*, pages 91–100, Vilamoura, Algarve, Portugal.

Teixeira, A. J. S., Pereira, C., e Silva, M. O., Alvarelhão, J., Neves, A. J. R., and Pacheco, O. (2011b). Output Matters! Adaptable Multimodal Output for New Telerehabilitation Services for the Elderly. In Dias, M. S., editor, *AAL 2011 – Proceedings of the 1st International Living Usability Lab Workshop on AAL Latest Solutions, Trends and Applications (In conjuction with BIOSTEC 2011)*, pages 23–35, Rome, Italy.

Theune, M., Klabbers, E., de Pijper, J., Krahmer, E., and Odijk, J. (2001). From data to speech: a general approach. *Natural Language Engineering*, 7(1):47–86.

Tran, K. and Popowich, F. (2016). Automatic Tweet Generation From Traffic Incident Data. In Gardent, C. and Gangemi, A., editors, *Proceedings of the 2nd*

International Workshop on Natural Language Generation and the Semantic Web (WebNLG 2016), Edinburgh, Scotland. Association for Computational Linguistics (ACL).

Turing, A. M. (1950). COMPUTING MACHINERY AND INTELLIGENCE. *Mind*, 49(49):433–460.

Turner, R., Sripada, S., Reiter, E., and Davy, I. P. (2006). Generating spatio-temporal descriptions in pollen forecasts. In *Proceedings of the Eleventh Conference of the European Chapter of the Association for Computational Linguistics: Posters & Demonstrations.*, pages 163–166, Trento, Itália. Association for Computational Linguistics.

University of Stanford (2015). The Stanford Parser: A statistical parser. Online, `http://nlp.stanford.edu/software/lex-parser.shtml`. Visited on Februay 14, 2015.

Vaudry, P.-L. and Lapalme, G. (2013). Adapting SimpleNLG for bilingual English-French realisation. In *Proceedings of the 14th European Workshop on Natural Language Generation (ENLG 2013)*, pages 183–187, Sofia, Bulgaria. Association for Computational Linguistics.

Vogiatzis, D., Galanis, D., Karkaletsis, V., Androutsopoulos, I., and Spyropoulos, C. D. (2008). A Conversant Robotic Guide to Art Collections. In *Proceedings of the 2nd workshop on Language Technology for Cultural Heritage Data, Language Resources and Evaluation Conference (LREC 2008)*, Marrakech, Morocco.

Weka (2017). ARFF (stable version). Online, `http://weka.wikispaces.com/ARFF`. Visited on January 16, 2017.

White, M. and Caldwell, T. (1998a). Beyond XSL: Generating XML-Annotated Texts with EXEMPLARS. Technical report, CoGenTex Technical Report, Ithaca, New York, USA.

White, M. and Caldwell, T. (1998b). EXEMPLARS: A Practical, Extensible Framework for Dynamic Text Generation. In *Proceedings of the Nineth International Workshop on Natural Language Generation*, pages 266–275, Niagara-on-the-Lake, Ontario, Canada. Association for Computational Linguistics.

White, T. (2015). *Hadoop: The Definitive Guide, Fourth Edition.* O'Reilly Media, Inc., 1005 Gravenstein Highway North, Sebastopol, CA 95472, USA, 4th edition.

Wilcock, G. (2001). Pipelines, Templates and Transformations: XML for Natural Language Generation. In *Proceedings of the First NLP and XML Workshop*, page 8, Tokyo, Japan.

Wilcock, G. (2005). An overview of shallow xml-based natural language generation. In Margit Langemets, P. P., editor, *Proceedings of Second Baltic Conference on HUMAN LANGUAGE TECHNOLOGIE*, pages 67–78, Tallinn, Estonia. Institute of the Estonian Language, Institute of Cybernetics, Tallinn University of Technology.

Witten, I. H. and Frank, E. (2005). *Data Mining: Practical Machine Learning Tools and Techniques*. Morgan Kaufmann Publishers, 1st edition.

Yu, J., Reiter, E., Hunter, J., and Sripada, S. (2003). SumTime-Turbine: A Knowledge-Based System to Communicate Gas Turbine Time-Series Data. In *Developments in Applied Artificial Intelligenc: Proceedings of IEA/AIE-2003*, pages 379–384, Loughborough, UK. Springer (LNAI 2718).