

**FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO**

# **Statistical Language Models applied to News Generation**

**João Ricardo Pintas Soares**



Mestrado Integrado em Engenharia Informática e Computação

Supervisor: Prof. Sérgio Sobral Nunes

July 24, 2017



# **Statistical Language Models applied to News Generation**

**João Ricardo Pintas Soares**

Mestrado Integrado em Engenharia Informática e Computação

July 24, 2017



# Abstract

Natural Language Generation (NLG) is a subfield of Artificial Intelligence. Its main goal is to produce understandable text in natural language, from a non-linguistic data input. Automated News Generation is a promising subject in the area of computational journalism which can use NLG to create tools that helps journalists in the news production, automating some steps. Most of these tools need a large amount of structured data as input and, for this reason, sports is a very natural subject to use because is a data-rich area. The automatization of steps, in the news production, brings benefits to journalists, namely the tools can summarize data and convert it into text. Then they just have to adjust it, making the process of production a lot faster. The need for this agile process was the main motivation of this dissertation.

The goal of this dissertation is to implement an Automated News Generation algorithm with the collaboration of ZOS, Lda. who owns the `zerozero.pt` project, an online media publisher with one of the largest football databases in the world. They will provide a dataset for exploration and research. Our approach is based on the use of Statistical Language Models to generate summaries from scratch, applying them to a system where the user can generate sentences about a specific match.

Our dataset is composed by summaries written about the season 2015/2016 of the Serie A of the Italian championship. After a manual analysis we decided that these summaries would be divided in four categories: introduction, goals, sent-offs and conclusion. Then we had to divide the summaries in sentences and assign each one to a category. The composition of the corpus is the following: 118 sentences for introduction, 93 for goals, 23 for sent-offs and 66 for conclusion. These categories were then divided in subtypes which correspond to the corpus used to train the N-gram-based models in order to produce sentences. We evaluate the final sentences regarding intelligibility and completeness, using one questionnaire. We concluded that, although these models generate sentences that have high scores in intelligibility and completeness, they also generate ones that scored poorly. We believe this is due to data sparsity. To try to overcome this problem, `zerozero.pt` provided an additional collection of summaries, so we can build a classifier to automatically classify each sentence in one of the four summaries categories. This way we were able to have more data to improve the models. The classifier returned all the sentences divided in the four categories and we decided to only use one of the categories dataset due to the lack of time. Introduction was the more appropriate since it is possible to delexicalize using the API data of each match. After delexicalization and merging the new corpus with the old introduction corpus we got 365 sentences for introduction, divided in subtypes that worked as the new training corpus. With the sentences generated from the new models we created a new questionnaire. Certain characteristics of each questionnaires sentence were analyzed and a comparison was made when it was pertinent.

**Keywords:** Automated Journalism; Computational Journalism; Natural Language Processing; Natural Language Generation; Statistical Language Models



# Resumo

Geração de Linguagem Natural (GLN) é um subcampo da Inteligência Artificial. O seu principal objetivo é produzir texto perceptível em linguagem natural, a partir de dados de entrada não linguísticos. Geração Automática de Notícias é um campo promissor na área de jornalismo computacional, que usa GLN para criar ferramentas que ajudam os jornalistas na produção de notícias, automatizando alguns passos. A maior parte destas ferramentas precisa de uma grande quantidade de dados estruturados como entrada e, por esta razão, desporto é um tema natural a abordar pois é uma área rica em dados. A automatização de passos, na produção de notícias, traz benefícios aos jornalistas, nomeadamente as ferramentas podem sumarizar dados e convertê-los em texto. Seguidamente apenas tem de ser ajustado, acelerando bastante o processo de produção. A necessidade de um processo mais rápido foi a principal motivação desta dissertação.

O objetivo desta dissertação é implementar um algoritmo de Geração Automática de Notícias com a colaboração da ZOS, Lda. que é proprietária do projeto zerozero.pt, um jornal online com uma das maiores bases de dados do mundo. O zerozero.pt vai fornecer um conjunto de dados para exploração e investigação. A nossa abordagem é baseada no uso de Modelos de Linguagem Estatísticos para gerar sumários de raiz, aplicando-os a um sistema onde o utilizador pode gerar frases relativas a um determinado jogo.

O nosso conjunto de dados é composto por sumários escritos acerca da temporada 2015/2016 da Serie A do campeonato Italiano. Após uma análise manual decidimos que os sumários seriam divididos em quatro categorias: introdução, golos, expulsões e conclusão. Seguidamente tivemos que dividir os sumários em frases e atribuir cada uma a uma categoria. A composição do corpus é a seguinte: 118 frases para introdução, 93 para golos, 23 para expulsões e 66 para conclusão. Depois estas categorias são divididas em subtipos que correspondem ao corpus usado para treinar os modelos baseados em N-gramas, de modo a produzir frases. Nós avaliamos as frases geradas de acordo com a inteligibilidade e a completitude, usando um questionário. Concluimos que, apesar dos modelos gerarem frases que tem boas classificações relativamente à inteligibilidade e completitude, também geram frases com classificações fracas. Acreditamos que isto se deve à escassez de dados. Para tentar superar este problema, zerozero.pt forneceu uma coleção adicional de sumários, de modo a que possamos implementar um classificador para automaticamente atribuir cada frase a uma das quatro categorias dos sumários. Desta forma conseguimos ter mais dados para melhorar os modelos. O classificador retornou todas as frases divididas nas quatro categorias e decidimos usar apenas uma delas devido à falta de tempo para terminar a dissertação. Escolhemos a introdução visto que é possível deslexicalizá-la usando os dados de cada jogo da API. Depois da deslexicalização e da junção do novo corpus ao antigo da introdução obtivemos 365 frases para introdução, divididas em subtipos que funcionaram como o novo corpus de treino. Com as frases geradas a partir dos novos modelos criamos um novo questionário. Determinadas características de cada frase dos questionários foram analisadas e comparadas quando pertinente.

**Palavras-Chave:** Geração de Linguagem Natural; Jornalismo Automatizado; Jornalismo Computacional; Processamento de Linguagem Natural; Modelos de Linguagem Estatísticos





# Acknowledgements

I would like to thank my supervisor, Prof. Sérgio Sobral Nunes, for suggesting this dissertation and for always being ready to help and guide me on the right path. I also want to thank José Devezas for his availability and knowledge that he passed to me.

I want to thank my parents and my brother for all the love and support through these five years and my whole life, and also for making this possible.

A special thanks to my girlfriend, Inês, for being my anchor, always believing in me and giving me the strength I needed to carry on.

Last but not least, I would like to thank my friends for the support, encouragement and for always being there for me.

João Ricardo Pintas Soares



*“Before anything else, preparation is the key to success.”*

Alexander Graham Bell



# Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
1.1	Context and Motivation . . . . .	1
1.2	Objectives . . . . .	2
1.3	Dissertation Structure . . . . .	2
<b>2</b>	<b>Natural Language Generation</b>	<b>5</b>
2.1	Historical Review . . . . .	5
2.2	Classification of NLG Systems . . . . .	6
2.2.1	Input into the System . . . . .	6
2.2.2	Communicative Goal of the System . . . . .	6
2.3	The Structure of an NLG System . . . . .	7
2.3.1	The Corpus . . . . .	7
2.3.2	NLG Architectures . . . . .	9
2.3.3	Intermediate Representations . . . . .	12
2.4	NLG Generic Approaches . . . . .	14
2.4.1	Knowledge-Based Approaches . . . . .	15
2.4.2	Statistical Approaches . . . . .	16
2.4.3	Hybrids Approaches . . . . .	17
2.5	Evaluation Methodologies . . . . .	17
2.6	NLG Tools . . . . .	18
2.6.1	Natural Language Toolkit (NLTK) . . . . .	18
2.6.2	NaturalOWL . . . . .	19
2.6.3	PyNLPI . . . . .	20
2.6.4	SimpleNLG . . . . .	20
2.6.5	OpenCCG . . . . .	20
2.6.6	SRILM . . . . .	20
<b>3</b>	<b>Statistical Language Modeling</b>	<b>23</b>
3.1	Problem Approach . . . . .	23
3.2	Overview . . . . .	24
3.2.1	The Chain Rule . . . . .	24
3.2.2	Markov Assumption . . . . .	25
3.2.3	N-Grams . . . . .	25
3.3	Input processing . . . . .	26
3.3.1	Data extraction . . . . .	26
3.3.2	Delexicalization . . . . .	27
3.3.3	Summaries typification . . . . .	31
3.3.4	Data preparation . . . . .	33

## CONTENTS

3.4	Language Modeling . . . . .	35
3.4.1	Training Language Models . . . . .	35
3.4.2	Sentences Generation . . . . .	37
3.5	Implemented System . . . . .	38
3.5.1	Description of the system . . . . .	38
3.5.2	Implementation of the system . . . . .	38
<b>4</b>	<b>Automatic Classification of Summaries</b>	<b>41</b>
4.1	Classifier . . . . .	41
4.1.1	Implementation and Classification . . . . .	41
4.2	Data preparation and Modelation . . . . .	42
<b>5</b>	<b>Evaluation</b>	<b>45</b>
5.1	Methodology . . . . .	45
5.2	Results and Discussion . . . . .	49
5.2.1	Evaluation of the implemented system's output . . . . .	49
<b>6</b>	<b>Conclusions and Future Work</b>	<b>57</b>
6.1	Summary . . . . .	57
6.2	Future Work . . . . .	58
<b>A</b>	<b>Questionnaire one</b>	<b>59</b>
<b>B</b>	<b>Questionnaire two</b>	<b>69</b>
	<b>References</b>	<b>79</b>

# List of Figures

2.1	Example of created messages (Source [RD97]). . . . .	10
2.2	Example of tree structure of Discourse Planning (Source [RD97]). . . . .	11
2.3	Architecture of a NLG System (Source [RD97]). . . . .	13
2.4	SPL representation (Source [RD97]). . . . .	14
3.1	Diagram explaining the process. . . . .	23
3.2	Information about Napoli x Sassuolo match <sup>1</sup> . . . . .	28
3.3	System's interface. . . . .	38
3.4	Part of the match information, provided by zerozero.pt API, in JSON. . . . .	39
5.1	Image of the questionnaire's structure. . . . .	48
5.2	Average classification per sentence on the first questionnaire. . . . .	49
5.3	Distribution of the sentences classification on the first questionnaire, for intelligibility. . . . .	50
5.4	Distribution of the sentences classification on the first questionnaire, for completeness. . . . .	51
5.5	Average classification per sentence on the second questionnaire. . . . .	51
5.6	Distribution of the sentences classification on the second questionnaire, for intelligibility. . . . .	52
5.7	Distribution of the sentences classification on the second questionnaire, for completeness. . . . .	53
5.8	Intelligibility and completeness average classification regarding the number of words per sentence on the first questionnaire. . . . .	54
5.9	Intelligibility and completeness average classification regarding the number of words per sentence on the second questionnaire. . . . .	54
5.10	Intelligibility and completeness average classification regarding the sentence's category on the first questionnaire. . . . .	55
5.11	Intelligibility and completeness average classification regarding the sentence's category on the second questionnaire. . . . .	56
5.12	Intelligibility and completeness average classification regarding the sentence's subtype on the second questionnaire. . . . .	56
A.1	Part 1 of questionnaire one. . . . .	59
A.2	Part 2 of questionnaire one. . . . .	60
A.3	Part 3 of questionnaire one. . . . .	61
A.4	Part 4 of questionnaire one. . . . .	62
A.5	Part 5 of questionnaire one. . . . .	63
A.6	Part 6 of questionnaire one. . . . .	64
A.7	Part 7 of questionnaire one. . . . .	65

## LIST OF FIGURES

A.8	Part 8 of questionnaire one. . . . .	66
A.9	Part 9 of questionnaire one. . . . .	67
A.10	Part 10 of questionnaire one. . . . .	68
B.1	Part 1 of questionnaire two. . . . .	69
B.2	Part 2 of questionnaire two. . . . .	70
B.3	Part 3 of questionnaire two. . . . .	71
B.4	Part 4 of questionnaire two. . . . .	72
B.5	Part 5 of questionnaire two. . . . .	73
B.6	Part 6 of questionnaire two. . . . .	74
B.7	Part 7 of questionnaire two. . . . .	75
B.8	Part 8 of questionnaire two. . . . .	76
B.9	Part 9 of questionnaire two. . . . .	77
B.10	Part 10 of questionnaire two. . . . .	78



# List of Tables

2.1	Overview of the NLG Tools. . . . .	22
3.1	Tokens used in delexicalization for category “introduction”. . . . .	29
3.2	Tokens used in delexicalization for category “goals”. . . . .	30
3.3	Tokens used in delexicalization for category “sent-offs”.. . . .	30
3.4	Tokens used in delexicalization for category “conclusion”. . . . .	31
3.5	Number of sentences per each subtype of introduction. . . . .	33
3.6	Number of sentences per each subtype of goals. . . . .	34
3.7	Number of sentences per each subtype of sent-offs. . . . .	34
3.8	Number of sentences per each subtype of conclusion. . . . .	35
4.1	Document-term matrix example. . . . .	42
4.2	Number of sentences per category. . . . .	43
4.3	Number of sentences per introduction’s subtype. . . . .	43
5.1	Corpus characterization. . . . .	46
5.2	Subtype of each set of sentences on questionnaire 1. . . . .	47
5.3	Subtype of each set of sentences on questionnaire 2. . . . .	47

## LIST OF TABLES

# Abbreviations

API	Application Programming Interface
CCG	Combinatory Categorical Grammar
D2T	Data-to-Text
FEUP	Faculty of Engineering of the University of Porto
FLM	Factored Language Model
LM	Language Model
MLE	Maximum Likelihood Estimation
MTT	Meaning-Text Theory
NLG	Natural Language Generation
NLP	Natural Language Processing
RST	Rhetorical Structure Theory
SFG	Systemic Functional Grammar
T2T	Text-to-Text
TAG	Tree Adjoining Grammars



# Chapter 1

## Introduction

In this chapter we present the context of this dissertation, the motivation and the objectives proposed.

### 1.1 Context and Motivation

Natural Language Generation is a Natural Language Processing task that converts computer based representations of data to text, in natural language. NLG techniques are commonly used to create systems that produce information easily understandable by a human. An example of how NLG systems can be used are to create Authoring Aids, systems used to generate routine documents, helping workers in their tasks. This is useful to them so they don't have to occupy their working time producing these documents when they could be working in their main task. As Reiter and Dale wrote [RD97]: “a computer programmer may spend as much time writing text (code documentation, program logic descriptions, code walkthrough reviews, progress reports, and so on) as writing code. Tools which help such people quickly produce good documents may considerably enhance both productivity and morale.”.

Automated News Generation is a promising subject in the area of computational journalism which can use NLG to create tools that helps journalists in the news production, automating some steps. Most of these tools need a large amount of structured data as input and, for this reason, sports is a very natural subject to use because is a data-rich area [Lev12]. The automatization of steps, in the news production, brings benefits to journalists, namely the tools can summarize data, converting it into text [Roo14]. Then they just have to adjust it, making the process of production a lot faster. The need for this agile process was the main motivation of this dissertation. It is also possible to use a NLG system to complement journalistic pieces with historical facts existing in the database.

The goal of this dissertation is to implement an Automated News Generation algorithm with the collaboration of ZOS, Lda. who owns the zerozero.pt project, an online media publisher with one of the largest football databases in the world. They have provided a dataset for exploration and research in this field.

In 2016, João Aires, student of FEUP, wrote a dissertation about this topic [Air16]. That was the first research work in this area with the collaboration of zerozero.pt, and had the same goal. His approach was to build a template-based system where he manually created templates for the different events and characteristics of a football match, based on the analysis of previous news. The results were good, having managed to have a fluid and perceptible text, in most of the cases. However, in some cases, the fluidity decreased because of the large amount of similar sentences, due to the use of templates. That made the summaries more propitious to repetitions of information and consequently seemed less natural. We tried a different method where the system learns with examples summaries, in order to generate sentences.

## 1.2 Objectives

The primary objective is to use Statistical Language Models to generate match summaries from scratch, applying them to a system where the user can generate a summary about a specific match. Zerozero.pt generates data of more than 6000 matches per week and produces news for an average of 100 games per week. It is expected that this system will facilitate the use this large amount of structured data and consequently increase the journalists productivity. The idea is to integrate this system in a tool where the journalist can select a game from a list and then generate the summary. With that summary the journalist just has to make some small adjustments and the journalistic piece is ready. This tool can work as an assistant and make the production process faster.

Comparing with the work done by João Aires, instead of building the templates manually and use a template-based approach it was decided that a method based in learning will be used. A corpora will be built based in a large amount of summaries about previous matches, and the system will use them to learn and produce models. Then when a specific match is selected to generate a summary, those models will be used to generate the final output.

In order to achieve these objectives the state of the art of NLG will be reviewed to determine which approaches and tools can be helpful to meet the requirements previously defined. Also, the evaluation methodologies will be analyzed to understand how to evaluate the quality of the summaries generated and so the results can be presented in this dissertation.

## 1.3 Dissertation Structure

Beyond this chapter this dissertation has five more chapters, giving a total of six chapters. In Chapter 2 we present a review of the state of the art regarding NLG. In the first section, we do a historical review about the subject. Section two describes how a NLG system can be classified. At Section three we explain the way a NLG system can be structured. The following section describes the most relevant approaches applied to NLG. Section five explains how the quality of a NLG system can be evaluated. Finally, at Section six, we present some tools found useful to apply in NLG. Chapter 3 focus on Statistical Language Models and how they are applied to the implemented system. In Section one we approach the problem and the process is explained. In

## Introduction

Section two we make an overview about the subject where it is briefly explained. Section three explains the whole procedure of input processing done to prepare the corpus. Language modeling explanation is presented in Section four. We describe the implemented system in Section five. Chapter 4 presents an approach to get more data. Section one describes how the classifier was implemented and then used. The preparation and modelation of the classified data is represented in Section two. Chapter 5 explains how the implemented system is evaluated. Section one explains the methodology adopted and in Section two the results are analyzed and discussed. In Chapter 6 we present the conclusions about the work done in this dissertation and the expectation for future work. Section one provides a summary and Section two describes the perspectives for future work.

## Introduction



## Chapter 2

# Natural Language Generation

### 2.1 Historical Review

NLG field has been in constant development. In the first approaches were produced systems that translated data in simple text, almost without variation or none at all, like advice giving systems [Swa77] or simple weather forecasts [KPG86]. Through the years the systems became more complex, with more linguistic insights and several methodologies were developed for the generated text be more variable [Air16].

In 1994 K.S. Jones suggested the division of phases of work in NLP in four phases: first between late 1940s and late 1960s, second from late 1960s to 1970s, third until the late 1980s and fourth from late 1980s onward [Jon01]. In the beginning NLP had a slow and painstaking progress, having difficulties with dictionary and grammar-based approaches, until the 70's. In the third phase, new ideas appeared about logic and formal semantics, what reanimated NLP, however this still proved to be too challenging to perform practical tasks at scale. Abram Hindle et al points the reason *"Both these approaches essentially dealt with NLP from first principles - addressing language, in all its rich theoretical glory, rather than examining corpora of actual utterances, i.e., what people actually write or say."* So in the 80's, a big step was given, switching to a corpus-based approach using statistically rigorous methods. This transition was possible with the large quantity of Natural Language text available online, with translations in multiple languages, and the growing of computational technology [HBS<sup>+</sup>12].

In the present, NLG can be considered as a consolidated field, based on how many systems were developed and in the amount of areas they were used. However, NLG is still a field open in many aspects and there is no unique and strictly defined method to face NLG problems [RSBB16]. The seek for systems that translate data into text has been increasing and NLG field tries to satisfy those real life demands, presenting practical applications of the systems [Air16]. Generating reports about weather from meteorological data in multiple languages [GDK94] and generating custom letters to answer user's questions [Coc96] are examples of practical applications.

Several companies tried to enter in this field but two stood out, Narrative Science<sup>1</sup> and Automated Insights<sup>2</sup>, both based in the United States of America. These companies created systems capable of turning structured data into text very quickly like generating news about sports leagues, creating financial reports and about other subjects that can have organized data [Roo14].

## 2.2 Classification of NLG Systems

A NLG system can be classified considering different criteria. M. Vicente et al. acknowledge two main elements that are essential to distinguish NLG systems[VB<sup>+</sup>15]:

- The input into the system
- The communicative goal of the system

### 2.2.1 Input into the System

The NLG systems can be distinguished in two types, depending on their input, Data-to-Text (D2T) and Text-to-Text (T2T). While the input of D2T is a set of data that don't form a text, like numerical data representing meteorological information, in T2T the output is obtained by extracting the most relevant information from the input text.

**Data-to-Text:** The type of input data can vary a lot. The most common systems use numerical data as input (e.g. information from sensors, medical equipment), however other structured data like labeled corpus, databases, knowledge bases or log archives are also used. Some authors refers to non-linguistic data as concept, so these systems are also known as Concept-to-Text. An example of a D2T system is GoalGetter [TKdP<sup>+</sup>01], that generates spoken reports of football matches in Dutch. Another example is Proteus [Dav74], which generates a summary of a *3 in a row* game from a list of movements.

**Text-to-Text:** This systems has as input texts or isolated sentences. There is a large number of systems that use T2T systems to generate summaries or simplified texts. Sauper and Barzilay designed a system which generates Wikipedia articles from a set of Internet documents. The structure of the article is determined by its subject [SB09].

### 2.2.2 Communicative Goal of the System

As previously noted NLG systems can be distinguished regarding the communicative goal of the system. The most significant are:

**Informative Texts:** The main goal of this system is to extract information from factual data. FoG [GDK94] and SumTime[RSH<sup>+</sup>05] are two examples of this kind of systems, which take numerical data as input to create weather forecasts. Another example is

---

<sup>1</sup><https://www.narrativescience.com/>

<sup>2</sup><https://automatedinsights.com/>

SkillSum[WR08], a tool that generates reports about academic evaluations, to help people with poor arithmetic and literacy knowledge.

**Summaries:** Textual summaries are used to generate a succinct version of one or more data sources. They can be related with various fields of work: medical[PRG<sup>+</sup>09], engineering [YRHM07], financial [Kuk83], sports [RM96], patents[MW08] among others.

**Simplified Texts:** The purpose of simplified texts is to help people with cognitive difficulties or some disability in reading. These texts are used in systems that produce texts for aphasic people [RTA<sup>+</sup>09] or systems that allows people with visual disabilities to examine graphics[FPRL06].

**Persuasive Texts:** Systems that try to influence or take advantage of the emotional state of the user. An example is STOP [RRO03], a system that generates smoking cessation letters to persuade users to quit smoking.

**Dialogue Systems:** These systems are used to improve the communication between humans and machines. User interacts with the system, which generates sentences, in natural language, conditioned by the immediately previous context. Many systems of this kind were developed like Beetle II[DIB<sup>+</sup>11] a tutorial system to increase knowledge on certain subjects, through dialogue, and GIVE software [KBC<sup>+</sup>09], a system used in virtual environments of games.

**Explanations:** The output of this kind of systems is an explanation of a sequence of steps that the system followed to execute an algorithm, process a transaction or solve a mathematical problem. P.Rex [Fie05], an example of this type of systems, is a tool of explanation of theorems.

**Recommendations:** The objective of this systems is to create recommendations processing and translating the data with the user's preferences and opinions. A system who use this type of systems is Shed [LCFQY14], that based on the user's profile and information in the Web, recommends custom diets.

## 2.3 The Structure of an NLG System

### 2.3.1 The Corpus

#### 2.3.1.1 Corpus-Based Approach

In this subsection will be explained the Corpus-Based Approach [RD97]. A Corpus is the collection of example inputs and their output which is a very good resource for the NLG system to learn.

Initially it is created the Initial Corpus and, where appropriate, their associated inputs, the data about the subject, using examples of news and articles written by humans, the output text. The corpus should contain all the details that are expected to be produced by the NLG system including

uncommon cases as well as the regular ones. If there is no text, about the subject, written by humans, it should be asked to experts about the subject to write examples of appropriate output text.

Then it is possible that the developer will need to change the initial corpus for a various number of reasons:

- The input data may not be available in the detailed and organized way that is needed when using NLG systems;
- The text may appear to be non-optimal for someone who is an expert about the subject;
- Different opinions between experts who may propose different outputs, for the same input data, enter in conflict. This can be solved asking them as a group so they can decide which is the best output option for a given input.

This changes will alter the Corpus, which will be referred as Target Text Corpus. To create it the developer needs to identify the parts of the initial Corpus that were written by a human with information that the NLG system can't access because it is not in the input data.

Reiter and Dale presents a simple example based on a railway station information system as Initial Corpus of the output text: *"There are 20 trains each day from Aberdeen to Glasgow. The next train is the Caledonian Express; it leaves Aberdeen at 10am. It is due to arrive in Glasgow at 1pm, but arrival may be slightly delayed because of snow on the track near Stirling. Thank you for considering rail travel."* answering the question: *"When is the next train to Glasgow?"*. Assuming that the input data has all the information needed to answer the question, like train scheduling with a list of departure and arrival times of each train in the network and individual info about it, each sentence of the content of the Initial Corpus can be classified as follows:

**Unchanging Text:** *"Thank you for considering rail travel."*, text that is always present in the output.

**Directly-Available Data:** *"The next train is the Caledonian Express, it leaves Aberdeen at 10am, it is due to arrive in Glasgow at 1pm and arrival may be slightly delayed."*, text with the information provided by the input data.

**Computable Data:** *"There are 20 trains each day from Aberdeen to Glasgow."*, text with information provided by the input data, like the directly-available data, but it is computed by the system, for example counting records about train travels.

**Unavailable Data:** *"because of snow on the track near Stirling."*, text about something that is not available in the input data.

Unchanging text parts are easy to generate by the NLG system because they can be manually inserted as strings. The Directly-Available Data can present some difficulties for the NLG system in making it readable for a human but it's easily accessible. For the Computable Data it is needed to decide if the results are worth the cost involved. Unavailable Data is the one that causes more problems for the NLG system. This texts are not possible to generate because the system can't

include data in its output if it is not in its input. This is something that is usually based on humans opinion or facts that himself can relate, can be something more personal added to the text or some fact not directly linked with the data. There are some ways to work around the problem like inserting more information in the system, eliminate the parts of Unavailable Data from the Corpus or expect the humans add that data to the final output text, if the system will work like an assistant to them.

### 2.3.1.2 Data-driven approaches to extend the Corpus Automatically

Building a corpus manually is a task that brings many difficulties, such as the highly need of human resources and time. This type of corpus have commonly a small size and little variation. When the data-driven approaches started to appear in NLG sphere, the automation in the creation and extension of the corpus became possible, bringing a solution that was needed with urgency. This way is possible to have a more varied output.

Manishina et al. propose the next extension solutions [MJHL16] so it can be possible to build a larger corpus, with more variety:

**Extending the system vocabulary with automatically obtained synonyms:** Aims to generate new variations of the existing sentences with the replacement of open class words (e.g nouns, verbs, adjectives) to their synonyms obtained automatically;

**Diversifying the set of syntactic structures by introducing multi-word paraphrases:** Replaces sub-phrases in a sentence with paraphrases that are acquired automatically;

**Making the system responses more 'human' and user-friendly by means of introducing a modal component:** In charge of giving an pragmatic/emotional side to the sentences created.

These solutions are completely automatic and just need initial methodology validation and assessment, executed by humans.

### 2.3.2 NLG Architectures

The main goal of NLG systems is to generate output text from input data, and this can be build in different ways. The most used architecture in NLG Systems is divided in three stages, according to Reiter and Dale [RD97]. These stages will be explained next.

#### 2.3.2.1 Text Planning

Text Planning is composed by two tasks, Content Determination and Discourse Planning, and is the initial stage of the NLG System.

1. **Content Determination:** In this task is determined which information will be outputted in text. A set of messages will be created from the data input, or other data sources. These are filtered and summarized by the system, in natural language, that labels and differentiate

them in Entities, Concepts and Relations within the subject of the text. Giving the example used in the explanation of the Corpus-Based Approach, specific trains, places and times can be considered Entities, the property of being the next train is a Concept, and departure and arrival can be relations between trains and times. In the Figure 1 is shown an example of messages that can be created in this phase.

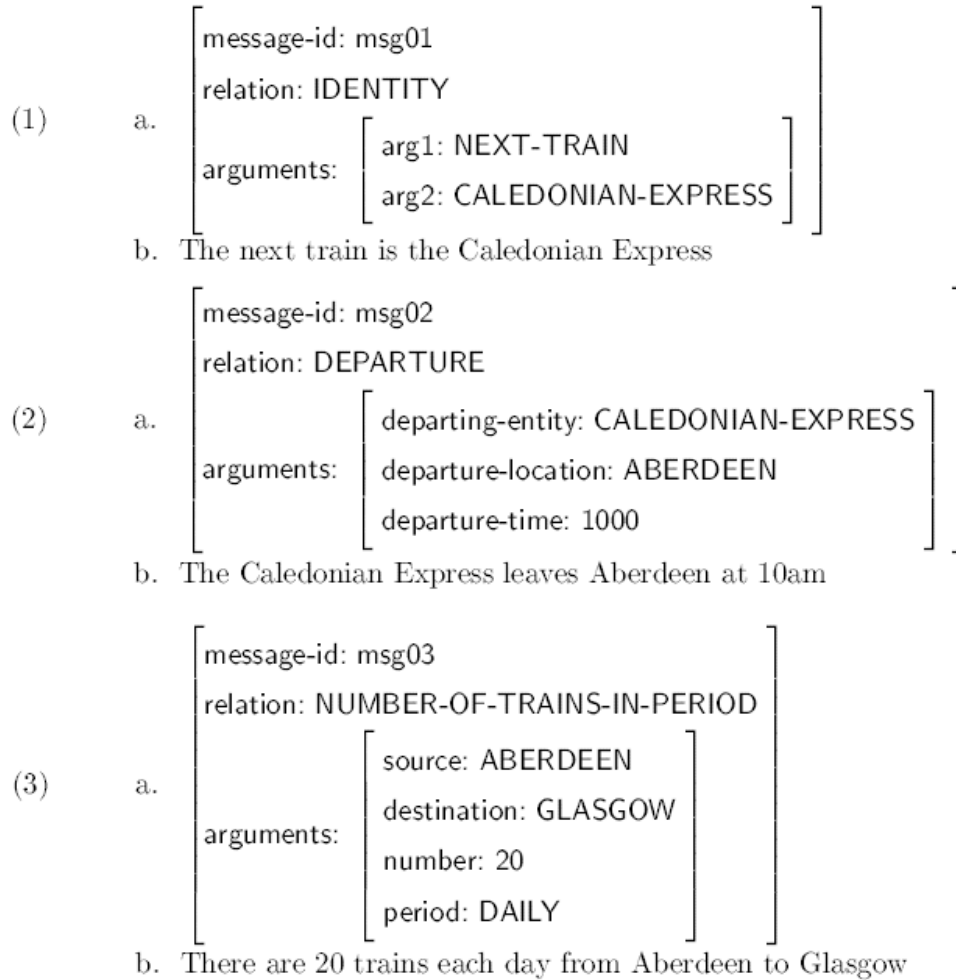


Figure 2.1: Example of created messages (Source [RD97]).

2. **Discourse Planning:** In this step the messages created in the previous step are structured and ordered to be presented as easily readable text. This has to be composed in a particular order, having a beginning, a middle and an end, in the simplest case. The structure will depend on how the developer wants the text to be composed. A good structure is fundamental for the text to be easy to read. The result of Discourse Planning is commonly represented as a tree structure like the one shown in Figure 2. The decisions made in the tree will have impact in the sentence aggregation and paragraph boundaries of the output text. The leaf nodes represent individual messages and the internal nodes display how messages are

grouped and related to each other and like explained by Reiter and Dale “*In some cases, the internal nodes also specify discourse relations between their children: in this example, the NUMBER-OF-TRAINS-IN-PERIOD and IDENTITY messages are placed in a sequence relationship, and the DEPARTURE message is an taken to be an elaboration of the IDENTITY message.*”

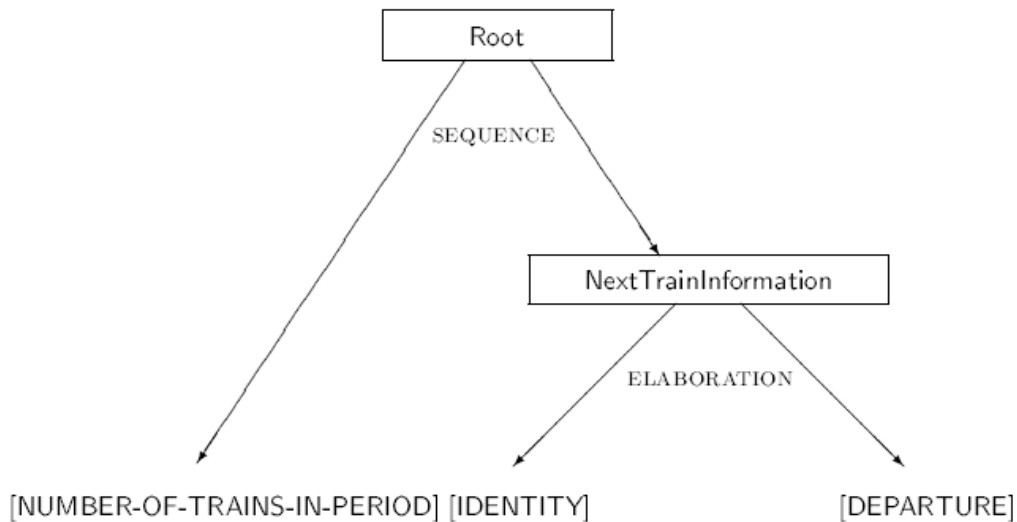


Figure 2.2: Example of tree structure of Discourse Planning (Source [RD97]).

## 2.3.2.2 Sentence Planning

In this stage the tasks executed are Aggregation, Lexicalization and Referring Expression Generation.

1. **Sentence Aggregation:** This is the process of grouping the messages, early created and planned, into sentences. In the example that is being used, Sentence Aggregation combine the IDENTITY and DEPARTURE messages into a sentence, “*The next train, which leaves at 10am, is the Caledonian Express*”. This step is not always required since there’s messages that can be conveyed in separate sentences. However aggregation is very useful to enhance fluency and readability of a text, it can be used to form paragraphs and other structures as well as sentences.
2. **Lexicalization:** In this phase it is decided which words and phrases shall be chosen to refer to the content shaped in the previous steps, based on the concepts about the text’s subject. Continuing with the railway station example, in this step is decided how DEPARTURE message should be represented: worlds like *leave* and *depart* are possibilities. Lexicalization can be done in different ways, like hard-coding a specific word or phrase for each concept

or relation, a DEPARTURE message can be always represented by the word *leave*. Other way is, when using a NLG system based in learning, vary words used to express a concept or relation, to have more variety or to accommodate subtle pragmatic distinctions. In this example *depart* can be more formal than *leave*.

3. **Referring Expression Generation:** This task is intimately related with Lexicalization since it is also concerned with the production of words and phrases to express the subject concepts, identifying its entities. In the text example of the Figure 1 is used the referring expressions “*the Caledonian Express*” and “*it*” referring to the subject entity CALEDONIAN-EXPRESS. However this task is used to avoid ambiguity, distinguishing one entity from other entities by finding distinct aspects of each one. This usually requires taking a look at the contextual factors, precisely the content of previous communications with the user, known as DISCOURSE HISTORY. Continuing to use the railway station example, deciding if *it* will be used to refer to CALEDONIAN-EXPRESS depends on what have been mentioned in the previous sentences.

### 2.3.2.3 Linguistic Realization

This stage executes the Linguistic Realization task.

1. **Linguistic Realization:** This final task is defined by Reiter and Dale as “*process of applying the rules of grammar to produce a text which is syntactically, morphologically, and orthographically correct.*” A Linguistic Realization process may choose to express the NUMBER-OF-TRAINS-IN-PERIOD message above as the sentence “*There are 20 trains each day from Aberdeen to Glasgow*”. The words *from* and *to* are added by the syntactic component of the realizer to indicate the train source and destination, in the sentence. The morphological component is responsible to transform the word *train*, in singular form, in its plural form *trains*. The orthographic component capitalized the first letter of the sentence and added a final dot in the end.

### 2.3.3 Intermediate Representations

Having decided the architecture of the NLG system, Reiter and Dale refer other important aspect, how the inputs and outputs should be represented, in the different stages. The internal representation between the three phases described above has to be specified, both from Text Planning to Sentence Planning and from Sentence Planning to Linguistic Realization. As for the initial input to the system, it will depend on if it's a D2T or a T2T system, on the other hand the final output will be text.

#### 2.3.3.1 Text Planning to Sentence Planning

The output of the Text Planning will be referred as Text Plan.



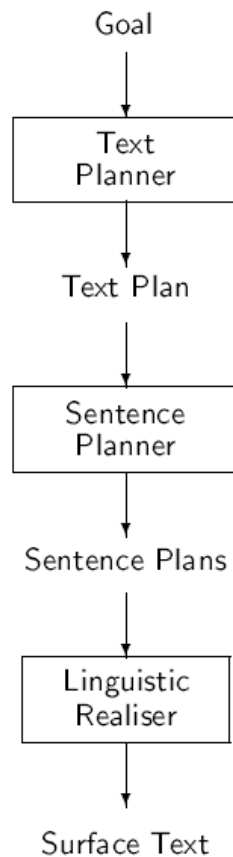


Figure 2.3: Architecture of a NLG System (Source [RD97]).

A text plan are usually represented as a tree. Each of its leaf nodes represent individual messages and each internal node show how messages are theoretically grouped together. The fact that they are grouped imposes constraints on the scope of subsequent sentence planning operations and also on possible locations for paragraph boundaries, in some cases. As noted above, on the description of Discourse Planing, the text plan may specify discourse relations between nodes. The most common approach of representing the messages, that compose the leaf nodes of the text plan, is to represent them in the same way they are represented in Sentence Planning. For this to be more clear Reiter and Dale give the following example: “*For example, if the NLG system uses templates for sentence plans, it might also use templates in the text plan leaf nodes. The templates in the text plan may then contain parameters represented by pointers into the domain knowledge base, whereas, when the sentence plans are constructed, these pointers will be replaced by words or phrases.*”.

## 2.3.3.2 Sentence Planning to Linguistic Realization

The output of the Sentence Planning will be referred as Sentence Plan.

One of the most common techniques used to represent sentence plans are called Abstract Sentential Representations. With an abstract representation language it's possible to specify the content words like nouns, verbs, adjectives and adverbs of a sentence and their relations. Sentence Planning Language (SPL) is one of the most used languages for sentence plans. Reiter and Dale explain this representation as: *“The SPL representation characterizes the sentence by means of a number of named attributes and their values, and allows values themselves to consist of named attributes and their values.* In Figure 4 is shown how the sentence *“There are 20 trains each day from Aberdeen to Glasgow”* is represented in SPL. With this language is possible to specify variations in the text in a very easy way. For example, the future-tense version of the example given before can be displayed by adding a single *(:tense future)* attribute-value pair to the SPL represented in Figure 4.

```
(S1/exist
  :object (01/train
    :cardinality 20
    :relations ((R1/period
      :value daily)
      (R2/source
        :value Aberdeen)
      (R3/destination
        :value Glasgow))))
```

Figure 2.4: SPL representation (Source [RD97]).

## 2.4 NLG Generic Approaches

In this section will be made an analysis to knowledge-based and statistical approaches, the most relevant methods applied to NLG. According to Vicente et al. [VBP<sup>+</sup>15] knowledge-based approaches use strong linguistic character resources like dictionaries, thesaurus, rules or templates. From these resources can be obtained syntactic, lexical and morphological information, among other types. On the other hand, on statistical approaches the information needed to generate natural language text from the input comes mostly from a corpus and the probabilities obtained from that corpus, that can be labeled or not. Comparatively with the knowledge-based approaches, statistical approaches are less domain or language restricted since, if the corpus is appropriate both in size and in content, it does not have to follow a set of rules or restrictions that may emerge because of a certain language or context characteristics. Generally, in NLG, the methods applied are based in the system's goal and these approaches are not exclusive, so hybrid approaches were created, combining the two introduced above. In the following subsections will be presented some relevant examples about these approaches, explained by Vicente et al. [VBP<sup>+</sup>15].

### 2.4.1 Knowledge-Based Approaches

Every knowledge-based system is capable to represent knowledge explicitly using resources such as ontologies, sets of rules or thesaurus. This is an aspect that all have in common. These systems are constituted by two subsystems: a knowledge base and an inference engine. The knowledge base is a database type for knowledge management, providing the necessary means for its collection, organization and recovery. The inference engine represents the reasoning part of the system, using the content of the knowledge base in a given sequence. This engine examines the knowledge base's rules one by one, and if one of them is met, a specific action is performed.

Knowledge-based approaches can still be related with template-based approaches. Template-based methods are used in NLG systems to map their non-linguistic input directly to the linguistic structures of the output sentences, defined as templates. These templates have variable values that are filled with the input data so it can produce the final output. Aires highlights a difference between this two approaches: *“The difference between knowledge-based and template-based approach resides in the fact that knowledge-based approaches require detailed grammatical and linguistic information to be embedded within the system.”* [Air16].

Some relevant examples about knowledge-based or hybrid approaches are:

**RST - Rhetorical Structure Theory:** RST [MT88] is one of the main theories applied to NLG and is related both with the cohesion of discourse and the structure of messages and paragraphs. The goal of this method is to recursively decompose any text into a set of elements that establish rhetorical or discursive relationships, known as schemas. In this set, the most relevant elements are called nucleus while the elements depending on them are referred to as satellites.

**SFG - Systemic Functional Grammar:** For systemic-functional linguistics, language is a resource that allows the production of meaning and is layered in three levels of abstraction: semantic, lexicographic and phonological/graphological. The way the communicative functions are expressed and how it affects the social dimension of language is described by SFG [HM85]. This theory takes to account three meaning dimensions: the propositional, the interpersonal, related with the type of speech, and the textual, the way information is structured and wrapped in a text.

**TAG - Tree Adjoining Grammars:** A TAG [JS97] is a lexicalized grammar composed by a set of elementary trees which incorporate semantic content. With this trees, using operations of substitution or union (adjoining), it is viable to produce a new labeled tree derived from the elementary ones. One of the advantages of using TAG is that it can solve, in the same task, the sentence planing and its realization, although that implies some loss of flexibility.

**MTT - Meaning-Text Theory:** MTT [ZM65] adopts a representation model that separates the semantic, syntactic, morphological and phonetic levels. In this model the NLG process consist in a continuous transformation of the representations through the referred levels. To perform intermediate conversions between levels are used equivalence rules.

### 2.4.2 Statistical Approaches

As stated in the introduction of this section, the statistical approaches are based in the extracted probabilities from a corpus. One of the most used tools for statistical approaches are the Language Models (LM) [CG96]. A statistical LM defines the language structure, it is a mechanism with a probability distribution which expresses the number of times that a sequence of  $n$  words  $P(w_1, w_2, \dots, w_n)$  appears in a set of texts. Therefore, a good LM can detect if a sentence is correctly built, from the probability associated to it. In the case of the probability associated is high it is said that the LM accepts the sentence. On the other hand, LM rejects it when the probability is low, this demonstrates that the sequence of words in the sentence does not belong to the set of texts on which the probability distribution was calculated.

A very interesting aspect for NLG is that a good LM can predict the way an input, or part of it, will be converted inside the system. The corpus is an important factor to take into account to determine the quality of the LM, more specifically the size of the corpus and the data source from which it is trained, since the number of contexts that a word can be used or the domain's amplitude to which the LM can be applied will be proportional to the dimension of the learning corpus.

The most used LM in NLG are the following:

**N-Grams:** A n-gram [BDM<sup>+</sup>92] is a subsequence of  $n$  elements of a given sequence. This LM is a probabilistic model that predicts the next element of a sequence in the form of a  $n - 1$  order, using the Markov1 chain of order. N-grams can be easily implemented and they are extensively used in recognition and learning algorithms. An aspect to be taken into account is that an n-gram model is very general, being necessary to adapt it to each application.

**Models based on Stochastics Grammars:** In Stochastics Grammars [Bod93] each grammatical rule has an associated probability, so the final result of the rules application produces a probability derived from that application. This type of LM presents the language constraints naturally. They also allow model dependencies while they are desired, although the definition of the models and their parameters may present difficulties in complex tasks.

**FLM - Factored Language Model:** Factored Language Model [BK03] is an extension of LM. In FLM each word is considered as a vector of  $k$  factors  $w_t = \{f_t^1, f_t^2, \dots, f_t^K\}$ . These factors can represent morphological classes, roots, or any lexical, syntactic or semantic characteristics. This model provides a probabilistic model  $P(f|f_1, \dots, f_N)$  where the prediction of a factor  $f$  is based on  $N$  parents  $\{f_1, \dots, f_N\}$ . Given an example, if  $w$  represents a word token and  $t$  represents a grammatical category (POS: Part-Of-Speech), the expression  $P(w|w_{i-2}, w_{i-1}, t_{i-1})$  presents a model to predict the current word token based on a conventional n-gram model as well as on the POS of the previous word. FLM allows users to specify relations between word tokens and POS, which is a great advantage.

### 2.4.3 Hybrids Approaches

Hybrid approaches combine the two approaches presented in the previous subsections, knowledge-based and statistical approaches. There are some systems who use this type of approaches: Nitrogen [LK98], FERGUS [ABD00], FLIGHTS [WCM10]. FLIGHTS for example, uses multiple knowledge bases such as user models, domain models and discourse history to perform content selection that should appear in the output. Next, the output is organized according to a template and the final text is produced by a tool known as OpenCCG [Whi12], used to apply n-grams and FLM, internally.

## 2.5 Evaluation Methodologies

Evaluation is very important in NLG systems because there is a need to know if the system is reliable, if the objectives are met and its potential utility. The evaluation of a system performance is done using different criteria, intrinsic and extrinsic [GJ93]. Intrinsic evaluation is related with the quality of the generated text, which is evaluated based on a set of criteria, commonly predefined. In turn, extrinsic evaluation is related with the system's role based on its setup purpose and the effect on human performance. Galliers and Jones give an understandable example about intrinsic and extrinsic evaluation: *"for a translation system, intrinsic criteria could be ones applying to the quality of the translation, and extrinsic criteria those applying to the ease with which post-editors could tweak these"*. Also evaluation can be distinguished in manual and automatic. In NLP field intrinsic and automatic evaluations are intimately connected so as extrinsic and manual evaluations. This is directly linked with their characteristics. Extrinsic evaluations has to be formulated in the context of a user task so it's hard to avoid human interference. In intrinsic evaluations only the system output must be considered so it's generally easier to develop automatic techniques [GJ93].

Generally manual evaluation produces good results, mainly if adequacy and fluency are the criteria used [Air16]. However it has limitations like being very expensive, slow and it can generate inconsistent results [GJ93]. An example of this type of evaluation was performed in the STOP system, when the developers performed inquiries to the users, to evaluate if the system met the objectives proposed. Regarding this evaluation cost and time length, it costed 75 thousand pounds and 20 months to be completed [RRO03].

Intrinsic evaluation, that values the characteristics of the system without consider external factors, generally requires the comparison of system's output with a reference text or corpus, using metrics or ratings. An NLG system can be evaluated as a whole or by modules and the communicative goal, discourse structure, coherence, ambiguity and vocabulary quality should be taken in to account when evaluating the system's output [Air16]. NLG communities have developed evaluation metrics, which are based on comparing output texts with a corpus of human-authoring texts and some of them are highly correlated with human opinions. Belz and Reiter enhance BLEU (BiLingual Evaluation Understudy) as one of the most successful metrics. It compares the generated text from the system with a set of texts authored by humans and evaluates how close the

generated text is from the set. BLEU scores from 0 to 1, only scoring the highest if the generated text is found in at least one of texts of the set [BR06].

Most of the NLG systems are evaluated using the following 3 intrinsic techniques [Bel09]:

1. Evaluation by trained assessors of the quality of the generated text according to different criteria, commonly using rating scales;
2. Automatic analysis of the similarity between system output and reference text, for example using BLEU, described above;
3. In this final step instead of automatic analysis is used human assessment about the similarity degree between system output and reference text.

## 2.6 NLG Tools

There is a large number of tools that can be applied in the different stages of a NLG system. In this section will be presented some of those tools found useful. In the case of NLTK and SRILM, there will be a in-depth explanation because they seem to be the most suitable tools to achieve the objectives.

### 2.6.1 Natural Language Toolkit (NLTK)

#### 2.6.1.1 Overview

Natural Language Toolkit is a collection of open-source program modules, tutorials and data sets, written in Python, very useful to work with human language data [LB02]. It was created by Steven Bird and Edward Loper in 2001.

This toolkit provides easy-to-use interfaces to over 50 corpora and lexical resources as WordNet, and aims to help the user with different tasks of NLP, like accessing corpora, processing strings, part-of-speech tagging, parsing and semantic reasoning [BKL09].

NLTK is a community-driven platform, very successful, having a great and understandable documentation. The creator wrote the book “*Natural Language Processing with Python*” cited above, which provides a practical introduction to all the tasks that can be used.

The source code of this software is distributed under the terms of the Apache Licence Version 2.0. NLTK is in continuous development so the existing modules are improved and news are created. The current version is 3.0.

In conclusion, this toolkit provides a simple and extensible framework, well documented and easy to learn, helping the user with several tasks of NLP.

### 2.6.1.2 Work done in Portuguese

NLTK provides a documentation of examples for portuguese processing<sup>1</sup>. In this examples are included texts written by Machado de Assis and includes the “Floresta Sinta(c)tica Corpus<sup>2</sup>”. This documentation shows that exists the possibility of:

- Search specific words or sequences in a text;
- Generate random text based on a given one;
- Sort the vocabulary of a sentence;
- Examine the relative and absolute frequency of words in a text;
- Find the most common ngrams which contain a target word;
- Get the tagged sentence data and train them;
- Segmentate sentences;
- Generate concordance for specific words;

However NLTK is more targeted to the english language, therefore, the works in portuguese are still very primitive.

### 2.6.2 NaturalOWL

NaturalOWL is an open-source natural language generator for OWL ontologies that supports English. It’s a D2T tool, written in Java, that evidences the benefits of using NLG techniques in the Semantic Web [GA07]. Bechhofer et al. defines OWL as follows: “*The Web Ontology Language OWL is a semantic markup language for publishing and sharing ontologies on the World Wide Web. OWL is developed as a vocabulary extension of RDF (the Resource Description Framework) and is derived from the DAML+OIL Web Ontology Language.*” [Bec09].

NaturalOWL uses the three stages of the NLG architecture explained in this dissertation, Text Planning, Sentence Planning and Linguistic Realization. Aires explain how the tool works in each of this stages as follows: “*NaturalOWL selects from the ontology all the logical facts that are directly relevant to that instance. The tool may be instructed to include facts that are further away in a graph representation of the ontology, up to a maximum (configurable) distance. The selected facts of distance one are then ordered by consulting ordering annotations which specify a partial order of properties. Second distance facts are always placed right after the corresponding directly relevant facts. In the application domains, this ordering scheme was adequate, although in other domains more elaborate text planning approaches may be needed. In the sentence planning stage, NaturalOWL lets the user to configure the maximum number of sentences that can be aggregated. Generally, NLG systems aggregate the maximum possible sentences in order to achieve better legibility. Finally, in realization stage, NaturalOWL takes the sentence planning output and represents it by adding punctuation symbols and capital letters where necessary.*” [Air16].

<sup>1</sup>[http://www.nltk.org/howto/portuguese\\_en.html](http://www.nltk.org/howto/portuguese_en.html)

<sup>2</sup><http://www.linguateca.pt/Floresta/>

### 2.6.3 PyNLPI

PyNLPI is a Python library which contains numerous modules useful to work with NLP tasks. It can be used to extract n-grams and frequency lists, and to build simple models. Furthermore it includes parsers for file formats common in NLP (e.g. FoLiA/Giza/Moses/ARPA/Timbl/CQL). Its most remarkable feature is a very extensive library to work with FoLiA XML (Format for Linguistic Annotation) [Gom10].

### 2.6.4 SimpleNLG

SimpleNLG is a Java library that helps the user in the Linguistic Realization stage [GR09]. This tool can be used to generate grammatically correct sentences, in English, and is well documented. It provides an interface to interact with the way sentences are built and combined, lexical and syntactic operations and linearization, so it can be able to construct a syntactic structure and linearizing it as text.

SimpleNLG was designed based in three criteria, flexibility, robustness and clear distinction between morphological and syntactic operations. It has an option of combining canned and non-canned strings to provides a more comprehensive syntactic coverage, increasing the flexibility of the system. Robustness is another concern of this tool so when an input are incorrect or incomplete it will not crash and still be able to produce a result, even that it is likely that the output is not the appropriate. Morphological and syntactic operations are well distinguished because the lexical component of the library is distinct from the syntactical component.

### 2.6.5 OpenCCG

OpenCCG is an open source NLP library, written in Java, which provides parsing and realization assistance with Combinatory Categorical Grammar (CCG) [SB11]. CCG is a lexicalized grammar formalism in which the grammatical elements are differentiated by a syntactic type or category of their inputs. The categories are related with the semantic type of the linguistic expression. Aires explains the procedure of OpenCCG as “*The OpenCCG realizer takes as input a logical form specifying the propositional meaning of a sentence, and returns one or more surface strings that express this meaning according to the lexicon and grammar.*” [Air16]. OpenCCG can applies n-grams and FLM, internally.

### 2.6.6 SRILM

SRILM is a collection of C++ libraries, executable programs and helper scripts, for building and applying statistical language models, which can be used for speech recognition and machine translation. It is also possible to create and evaluate language models based on N-gram statistics and supports related tasks like statistical tagging and manipulation of N-best lists and word lattices [S<sup>+</sup>02]. This toolkit is under development in the SRI Speech Technology and Research Laboratory since 1995 and it is for noncommercial use.



SRILM provides an extensible set of LM classes, non-standard LM types, including tagging and N-best rescoring. This toolkit grew out of the desire to build a more efficient implementation of the LM algorithms; to achieve flexibility and extendibility so it can ease the research into new types of LMs and, at the same time, the existing components can be reused; to provide a clean design with both an API and a toolbox of commands for LM handling. Its main goal is to support LM estimation and evaluation. Estimation is the creation of models from the training data. Evaluation is the computation of the probability of a text corpus.

### 2.6.6.1 Work done in Portuguese

Miranda de Novais and Paraboni implemented a shallow surface realization system which uses FLMs of Portuguese and applied it to the generation of Brazilian newspaper headlines [dNP13]. This system will be briefly explained. The authors approach surface realization which is a task of mapping abstract sentence representations to output text, a sequence of words in the target language. In this task the input is, commonly, language-independent representation of the sentence's content.

Miranda de Novais and Paraboni focused their work on generate-and-select NLG architecture, introduced in [Lan00]. This type of systems outputs text from an abstract representation as input by generate a large amount of alternative surface realizations and thereafter choose the most likely sequence of words with the help of a statistical LM. It has many advantages related with the fact that is a statistical approach, namely low cost development, as it does not require corpus annotation, and language independency. Nevertheless, it also has a disadvantage, which is the need of a large data set to train the system, so it can compensate the data sparseness. Data sparseness is specifically critical using morphologically rich languages like Portuguese. A Portuguese LM require a much larger training dataset than an English one, to achieve results that can be comparable [dNPF11]. To beat these difficulties, the authors used FLMs in the development of the shallow surface realization system, which was applied to the generation of newspapers headlines in Portuguese.

Tools	Language	Licence	Last Update
NLTK	Python	Apache License Version 2.0	March 2017
NaturalOWL	Java	GNU Library Public License	April 2013
PyNLPI	Python	GNU Library Public License	February 2017
SimpleNLG	Java	Mozilla Public License	August 2016
OpenCCG	Java	GNU Library Public License	May 2016
SRILM	C++	SRILM Research Community License	November 2016

Table 2.1: Overview of the NLG Tools.

## Chapter 3

# Statistical Language Modeling

### 3.1 Problem Approach

As established in Chapter 1 the main objective of this dissertation is to generate match summaries from scratch, in Portuguese, making it easier to use the large amount of structured data associated with a football match. Integrating this into a system where the user can select a specific match will enable the generation of a set of sentences that cover all or most of the data. This system may work as a journalists assistant, speeding up the news production process. To achieve these goals, Statistical Language Models will be used to generate phrases about the events of a football match. The tool that will be used to work with Statistical LMs is the SRILM toolkit (Subsection 2.6.6). It has the ability to generate models from a training corpus and then generate sentences from those models. In Figure 3.1 is presented a diagram describing the whole process. First we select from where the summaries will be extracted. Knowing that, we have to process the data that will serve as input to the creation of models. In this task we extract the summaries and delexicalize them. After that the summaries are divided in categories and then those categories are divided in subtypes, building the corpus. Having the corpus, the LMs are created and will work as input for the implemented system, which is responsible to generate the final sentences.

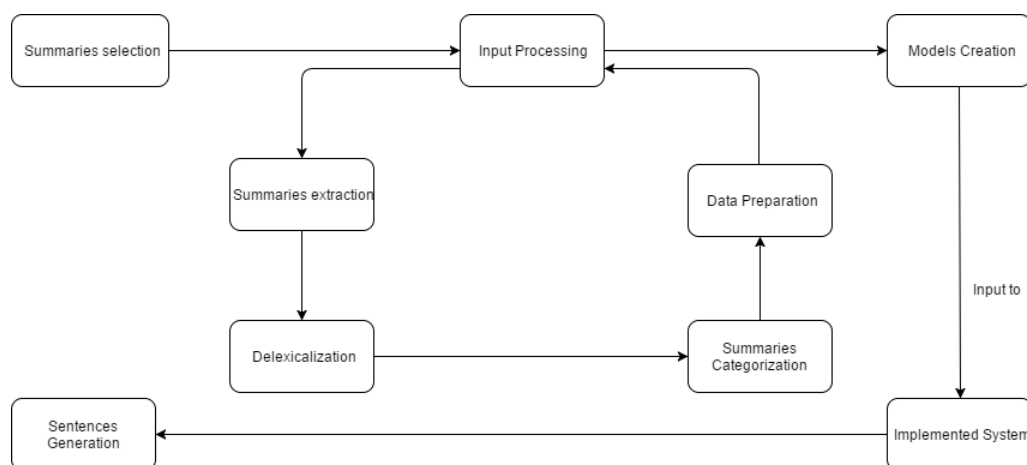


Figure 3.1: Diagram explaining the process.

## 3.2 Overview

All the general equations presented in this section are accordingly to Jurafsky and Martin [JJ00].

Statistical LMs are included in the statistical approaches, which are based in the extracted probabilities from a corpus, introduced in the Subsection 2.4.2. They are very useful in NLP applications with text as output, making possible to evaluate the likelihood of different sentences being generated based on a training corpus. Machine translation, speech recognition, information retrieval and POS tagging are some of many examples of these applications [S+02].

The main goal of a Statistical LMs is to compute the probability of word strings and can also be called as a stochastic process model for word sequences. It defines the language structure, expressing the number of times that a  $n$  words sequence “ $w_1, w_2, \dots, w_n$ ” appears in a set of texts. It can also be used to calculate the probability of the next word in a sequence:  $P(w_4|w_1, w_2, w_3)$ , this represents the probability of the word four appears after a 3-sequence word. Relative frequency counts is a way to evaluate these probabilities. As an example, the sentence “*Today I went to school*” will be used. Counting the number of times that “*Today I went to*” appears in a corpus and when it is followed by “*school*”, allows to estimate the conditional probability  $P(\text{school}|\text{Today}, I, \text{went}, \text{to})$ .

$$P(\text{school}|\text{Today}, I, \text{went}, \text{to}) = \frac{C(\text{Today } I \text{ went to school})}{C(\text{Today } I \text{ went to})} \quad (3.1)$$

To perform counts like this it is needed a large corpus. Jurafsky and Martin [JJ00] states that “*With a large enough corpus, such as the web, we can compute these counts and estimate the probability (...) While this method of estimating probabilities directly from counts works fine in many cases, it turns out that even the web isn’t big enough to give us good estimates in most cases. This is because language is creative; new sentences are created all the time, and we won’t always be able to count entire sentences.*”. At the same time, trying to calculate the joint probability of a sentence or sequence of words as “*Today I went to school*” would implicate to know how many sequences of five words, out of all possibles, would be “*Today I went to school*”. That would be a lot of data to evaluate. Knowing this, another methods need to be introduced, in order to facilitate the estimation of probabilities.

### 3.2.1 The Chain Rule

The Chain Rule, also known as the General Product Rule, decomposes the joint probability of a sentence or sequence of words, in conditional probabilities of the next word given his past, computing their product [Roa01]. Generally it can be represented as:

$$P(w_1, w_2, w_3, \dots, w_n) = P(w_1) * P(w_2|w_1) * P(w_3|w_1, w_2) * P(w_n|w_1, \dots, w_{n-1}) \quad (3.2)$$

Using the previously sentence example, it is represented as follows:

$$P(\text{"Today I went to school"}) = P(\text{Today}) * P(I|\text{Today}) * P(\text{went}|\text{Today}, I) * P(\text{to}|\text{Today}, I, \text{went}) * P(\text{school}|\text{Today}, I, \text{went}, \text{to}) \quad (3.3)$$

Even with this decomposition it is not possible to estimate the probability of the next word, when its set of past words is a large sequence. This happens for the same reasons previously stated.

### 3.2.2 Markov Assumption

To estimate the probabilities of the members of the Chain Rule, the Markov Assumption is commonly used. In text, the Markov Assumption can be translated as: The future does not depend on the past, given the present [LS99]. Instead of calculating  $P(\text{school}|\text{Today}, I, \text{went}, \text{to})$  it is possible to simplify to  $P(\text{school}|\text{to})$ , making the estimation of word's probability feasible.

$$P(\text{school}|\text{Today}, I, \text{went}, \text{to}) \approx P(\text{school}|\text{to}) \quad (3.4)$$

This is called a bigram model. However it is possible to derive the bigram to the trigram and therefore to the N-gram.

### 3.2.3 N-Grams

The general equation for a N-gram approximation to the next word's conditional probability is:

$$P(w_n|w_1^{n-1}) \approx P(w_n|w_{n-N+1}^{n-1}) \quad (3.5)$$

A N-gram is a model that designates probabilities to sequences of words [GR03]. The N in N-gram represents the number of words in that sequence. For example, a 2-gram, known as bigram, is a sequence of two words, such as "Today I", "I went", "went to" and a 3-gram, known as trigram, is a three-word sequence like "Today I went", "I went to", "went to school". To calculate the N-gram probabilities is used a method called Maximum Likelihood Estimation, also known as MLE. Exemplifying with the probability of a bigram model:  $P(w_n|w_{n-1})$ , the MLE is calculated by counting the number of times the word  $w_n$  appears after the previous word  $w_{n-1}$  and counting the number of bigrams that has  $w_{n-1}$  as first word. The counting of bigrams with  $w_{n-1}$  as first word can be simplified to a counting of unigrams, since  $C(w_{n-1}w) = C(w_{n-1})$  [JJ00].

$$P(w_n|w_{n-1}) = \frac{C(w_{n-1}w_n)}{C(w_{n-1})} \quad (3.6)$$

For this to be clear, an example will be presented, using a small corpus. This example was adapted from the one used in [JJ00]. First it is necessary to add the symbol  $\langle s \rangle$ , that represents the beginning of a sentence, and the  $\langle /s \rangle$ , representing the end. The symbol  $\langle s \rangle$  is relevant so it can be possible to estimate the probability of the first word bigram model. The  $\langle /s \rangle$  is responsible

to make the bigram grammar a true probability distribution.

*<s> I eat fruit to be healthy </s>*  
*<s> I ate vegetables on Monday </s>*  
*<s> Monday I went to school </s>*  
*<s> He trains to be faster </s>*

Some bigram probabilities can be calculated as follows:

$$\begin{array}{lll} P(I|<s>) = \frac{2}{4} & P(Monday|<s>) = \frac{1}{4} & P(</s>|Monday) = \frac{1}{2} \\ P(went|I) = \frac{1}{3} & P(school|to) = \frac{1}{3} & P(be|to) = \frac{2}{3} \end{array}$$

The general equation to calculate MLE in a N-gram is:

$$P(w_n|w_{n-N+1}^{n-1}) = \frac{C(w_{n-N+1}^{n-1}w_n)}{C(w_{n-N+1}^{n-1})} \quad (3.7)$$

### 3.3 Input processing

In order to be possible to generate Statistical LMs it is necessary to build a training corpus. This corpus will work as an input for the LMs. As stated previously, the primary objective is to generate sentences about the events of a football match. With the collaboration of zerozero.pt we have access to all the necessary data, both summaries and information on its website as also data from an API.

#### 3.3.1 Data extraction

To build a corpus we need news that summarizes a football match. Therefore a championship has to be chosen to enable the manual analysis of all the news that were written, about the matches of a specific season. The championship that we will analyze is the Italian one, known as Serie A, from 2015/2016 season<sup>1</sup>. It is composed by 20 teams, in which they play each other 2 times, in a total of 38 rounds. In each round there are 10 matches but only the matches with the top teams are reported. To extract the data from zerozero.pt website we manually check each of the 380 games of the Serie A to see if it was reported. Then we copy the content of the summary to a text file so it can be analyzed in the next step. An example of a summary is presented below, in Portuguese and in the respective English translation. The corresponding match is Napoli vs Sassuolo from the 20th round<sup>2</sup>.

<sup>1</sup>[https://www.zerozero.pt/edition.php?id\\_edicao=87949](https://www.zerozero.pt/edition.php?id_edicao=87949)

<sup>2</sup><https://www.zerozero.pt/news.php?id=168273>

**Portuguese**

O Napoli recebeu e venceu o Sassuolo por 3x1, este sábado, em encontro da 20.<sup>a</sup> jornada da Serie A. No San Paolo, os napolitanos estiveram a perder, valendo um golo de Diego Falcinelli, aos 3', de penákti, mas conseguiram a reviravolta. Jose Maria Callejon (19') e Gonzalo Higuaín (42' e 90') marcaram os tentos da formação de Maurizio Sarri. Com este resultado, o Napoli passa a somar 44 pontos e está na liderança, aumentando a vantagem, uma vez que o Inter empatou nesta ronda. O Sassuolo continua com 31 próximo dos lugares de acesso às provas europeias.

**Translation in English**

Napoli hosted and beat Sassuolo by 3x1, this Saturday, on the round 20 of Serie A. At San Paolo, the Napolitans were losing, with a penalty goal from Diego Falcinelli, at 3', but they completed the comeback. Jose Maria Callejon (19') and Gonzalo Higuaín (42' and 90') scored the goals for Maurizio Sarri's team. With this result, Napoli has 44 points and leads, increasing the advantage, since Inter tied in this round. Sassuolo continues with 31 near the places of access to the European competitions.

Figure 5 shows the match information regarding the previously summary's match.

In total, 153 summary news were analyzed, out of 380 games. As explained before, only the top teams matches are reported, what gives an average of 4 summaries per round, i.e. 4 reported matches per round, out of 10 possible.

**3.3.2 Delexicalization**

Delexicalization is the process of replacing words in a text, that refer to data that can be modified without removing the meaning of that text, by tokens or symbols representing the value of that word [WGK<sup>+</sup>15]. These word usually represent entities or values that can be replaced by other entities or values, in order to make sentences more variable.

Wen et al. [WGK<sup>+</sup>15] use delexicalization in the implementation of a spoken dialogue system to overcome the need of a well-labeled dataset to train the system. It is presented a statistical language generator that uses a recurrent neural network that is trained by a delexicalized corpus where each value is replaced by a token representing its corresponding slot.

As stated before, the primary objective is to generate sentences that translate data of any match into text. To accomplish this we need the training corpus to be general, i.e. with variable words that can assume a different value in different matches. All the entities that refers teams, players, competition, day of the week and the numerical values such as a minute of an event, the result, points and position of a team in the league table can be replaced with tokens representing the meaning of the value their replacing. If all the sentences in a corpus are delexicalized, the generated sentence will also be delexicalized. Exemplifying with the following sentence:

<sup>3</sup><http://www.zerozero.pt/match.php?id=5016626>

## Statistical Language Modeling

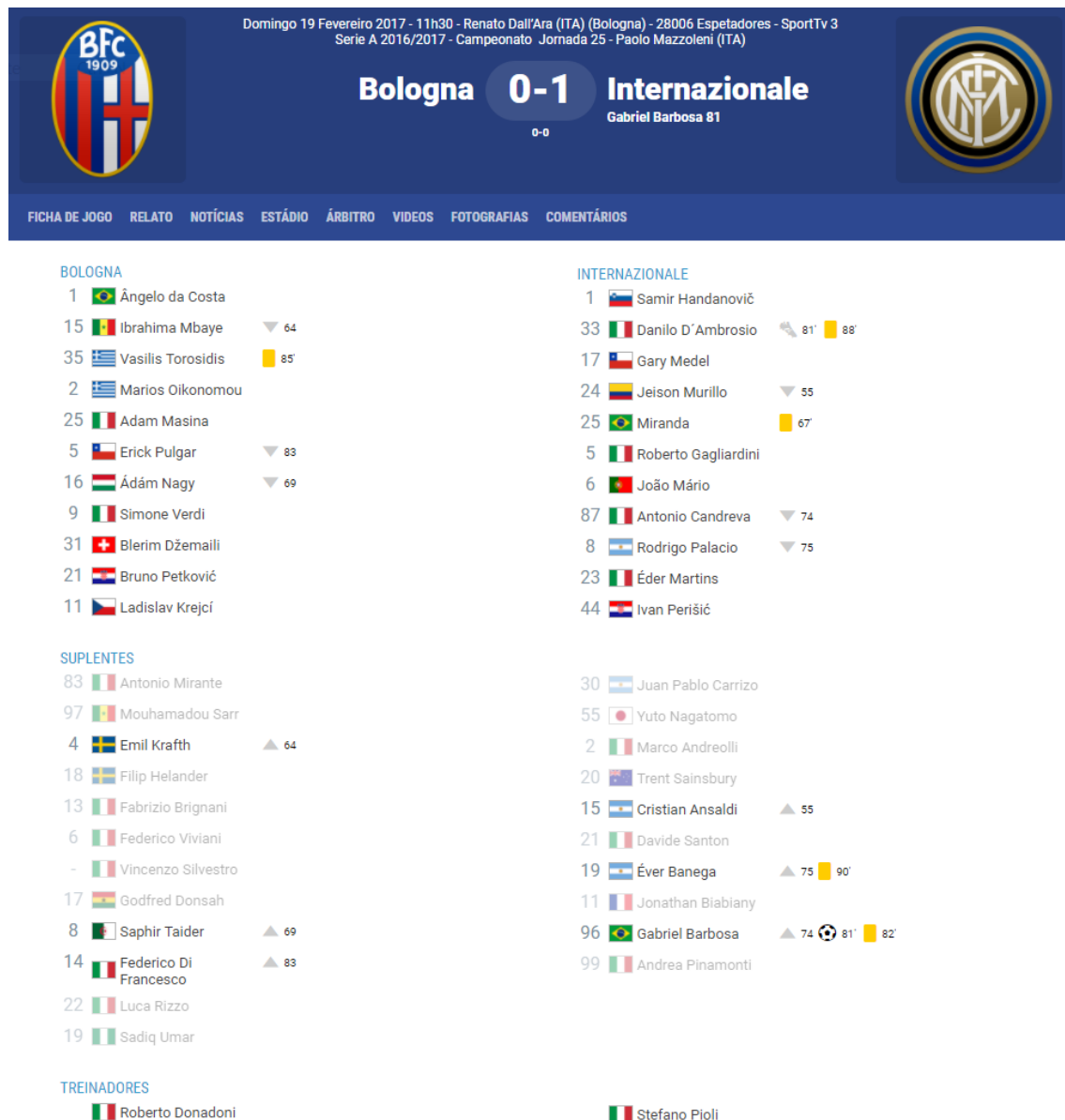


Figure 3.2: Information about Napoli x Sassuolo match<sup>3</sup>.

**Before delexicalization:** Napoli hosted and beat Sassuolo by 3x1, this Saturday, on the round 20 of Serie A.

**Delexicalized sentence:** <home:team> hosted and beat <away:team> by <home:team:result> x <away:team:result>, this <weekDay>, on the round <round> of <competition>.

This delexicalized sentence is now suitable to any match where the home team beat the away team. Now, to exemplify a case where we can't delexicalize all the entities or numerical values, we will use this sentence:

**Before delexicalization:** Mario Mandzukic, at 27 ', would make the tie for Juventus in a play that began in Buffon and had in Morata the assistant.



**Delexicalized sentence:** *<goalScorer:player>*, at *<goal:minute>* ', would make the tie for *<goalScorer:team>* in a play that began in Buffon and had in *<goalAssistant:player>* the assistant.

In this case we could not delexicalize the entity “*Buffon*” because we have no way to know who is the player that started the play, trough the provided data about the match. We can only tokenize the entities or numerical values that we will later have access over the API. The way we found to address this problem was to delete these entities and the words that link them to the sentences, so they remain well constructed, and the entities are not added to the LM’s vocabulary.

In this process is used a total of 44 different tokens, that are presented in the next tables. The decision of using these tokens was based in the analysis of all the summaries extracted. The words used to represent variable values, used in most of the sentences, like the home team or the away team, were replaced by tokens. We divide the tokens per category, so it is easier to understand their meaning. The tokens used for “introduction” are presented in Table 3.1.

<home:team>	Team that hosts the match.
<away:team>	Team playing as visitant.
<coach:home:team>	Coach of the home team.
<coach:away:team>	Coach of the away team.
<round>	Round of the match.
<competition>	Competition of the match.
<weekDay>	Day of the week.
<momentDay>	Moment of the day when the match starts (morning, afternoon, night).
<result:home:team>	Number of goals scored by home team in the end of the match.
<result:away:team>	Number of goals scored by away team in the end of the match.

Table 3.1: Tokens used in delexicalization for category “introduction”.

For “goals” the tokens used are presented in Table 3.2. It is important to refer that each goal sentence is just referred to one goal.

<team1>	We use this token when it is not specified any characteristic of the team (home, away, winning, losing). This token refers to the team that scored the goal described in the sentence.
<coach:team1>	Coach of the team referred as team 1 (see explanation of <team1> in this table).
<team2>	We use this token when it is not specified any characteristic of the team (home, away, winning, losing). This token refers to the team that conceded the goal described in the sentence.

<coach:team2>	Coach of the team referred as team 2 (see explanation of <team2> in this table).
<nGoals>	Number of goals scored for both teams at the moment that the token appears.
<currentResult:home:team>	Number of goals scored by the home team at the moment that the token appears.
<currentResult:away:team>	Number of goals scored by the away team at the moment that the token appears.
<goalScorer:player>	Player that scored the goal.
<goalScorer:minute>	Minute of the goal.
<goalAssistant:player>	Player that made the assistance for the goal.
<goalScorer:team>	Team of the player that scored the goal.
<goalConceded:team>	Team that conceded the goal.
<coach:goalScorer:team>	Coach of the team that scored the goal.
<coach:goalConceded:team>	Coach of the team that conceded the goal.
<goalkeeper:goalConceded:team>	Goalkeeper that conceded the goal.
<homeOrAway:goalScorer:team>	Saves if the scoring team played home or a away.
<currentNGoal>	Goals scored by both teams until the moment that the token appears.
<currentNGoal:goalScorer:team>	Goals scored by the team that scored the goal until the moment that the token appears.
<goalHalf>	Half where the goal happened.

Table 3.2: Tokens used in delexicalization for category “goals”.

The tokens used for “sent-offs” are presented in Table 3.3. It is necessary to refer that each sent-off sentence just refers to one sent-off.

<redCard:player>	Player that got expelled of the match.
<redCard:team>	Team of the player that got expelled.
<redCard:minute>	Minute of the player’s sent-off.
<coach:redCard:team>	Coach of the team that got the player expelled.
<noRedCard:team>	The opposing team from which the player was expelled.
<nPlayers:redCard:team>	Number of players of the team that got the player expelled, on the field until now.

Table 3.3: Tokens used in delexicalization for category “sent-offs”..

For “conclusion” the tokens used are presented in Table 3.4.

<win:team>	Team that won the match.
<lose:team>	Team who lost the match.
<points:between:teams>	Points difference between the home team and the away team and vice-versa.
<coach:win:team>	Coach of the winning team.
<coach:lose:team>	Coach of the losing team.
<team1>	The first team in a sentence, when it is not specified any characteristic of the team (home, away, winning, losing). This is only used when the final result is a draw and the order of appearing of the teams is not important, since they will win the same amount of points in the league table.
<points:team1>	Points of the team 1 (see explanation of <team1> in this table) in the league table.
<position:team1>	Position of the team 1 (see explanation of <team1> in this table) in the league table.
<coach:team1>	Coach of the team 1 (see explanation of <team1> in this table).
<team2>	The second team in a sentence, when it is not specified any characteristic of the team (home, away, winning, losing). This is only used when the final result is a draw and the order of appearing of the teams is not important, since they will win the same amount of points in the league table.
<points:team2>	Points of the team 2 (see explanation of <team2> in this table) in the league table.
<position:team2>	Position of the team 2 (see explanation of <team2> in this table) in the league table.
<coach:team2>	Coach of the team 2 (see explanation of <team2> in this table).

Table 3.4: Tokens used in delexicalization for category “conclusion”.

### 3.3.3 Summaries typification

In the process of extracting each summary, we have to read it to understand what it is about. In all the summaries we found a pattern: first the journalist introduces the game, then he talks about the goals and sent-offs, if there was any, and finalize it with a conclusion. Due to this reason we divided each summary in four categories: introduction, goals, sent-offs and conclusion.

The **introduction** opens the summary and presents the teams that played the match, the result and the competition and may also state the round of the game, the day of the week and the moment of the day.

In **goals** are included all the goals scored in the match, including the player who scored, its team, the minute of the goal and can also refer the player who made the assist, if he exists and the type of goal, i.e. if it was a penalty or an own-goal.

**Sent-offs** describes each event of the match where a player is expelled, including the player, its team, the minute and may also refer the number of players that stayed on the field of that team.

**Conclusion** closes the summary by stating how both teams stood in the league table, after the match, regarding their points and position.

The following summary corresponds to the match Bologna vs Internazionale from the 10th round<sup>4</sup>:

### Portuguese

O Inter foi ao terreno do Bologna vencer por uma bola a zero, esta terça-feira, em encontro da 10.<sup>a</sup> jornada da Serie A. No Estádio Renato Dell'Ara, em menos de 10 minutos (54' e 61'), Felipe Melo foi expulso por duplo amarelo. Mesmo a jogar com 10, o Inter chegou ao triunfo, valendo um golo de Mauro Icardi, aos 67 minutos. Com este resultado, os nerazzurri passam a somar 21 pontos e comandam, ainda que à condição, o campeonato. Já o Bologna tem seis e continua em zona de despromoção.

### Translation in English

Inter went to the Bologna field to win 1-0, this Tuesday, on a match of round 10 of Serie A. At the Renato Dell'Ara Stadium, in less than 10 minutes (54 'and 61'), Felipe Melo was sent off with a double yellow. Even playing with 10, Inter reached the victory, with a goal from Mauro Icardi, in the 67th minute. With this result, the Nerazzurri sums 21 points and command, provisionally, the championship. Bologna has six and is still in the relegation zone.

We classify this summary as follows:

**Introduction:** Inter went to the Bologna field to win 1-0, this Tuesday, on a match of round 10 of Serie A.

**Goals:** Even playing with 10, Inter reached the victory, with a goal from Mauro Icardi, in the 67th minute.

**Sent-offs:** At the Renato Dell'Ara Stadium, in less than 10 minutes (54 'and 61'), Felipe Melo was sent off with a double yellow.

**Conclusion:** With this result, the Nerazzurri sums 21 points and command, provisionally, the championship. Bologna has six and is still in the relegation zone.

The content of each text file is divided the same way this summary was classified. It has now four categories, each one with the sentences written about it. In the case of there was any sentence written about a category, the category stays empty.

---

<sup>4</sup><https://www.zerozero.pt/news.php?id=163774>

### 3.3.4 Data preparation

After we have all the data extracted and typified we have to prepare the data to build the training corpus for each LM. We have now four sets of sentences about each part of the summaries. Based on the content, introduction, goals, sent-offs and conclusion have to be divided into subtypes because the sentences can be specific. Each subtype will correspond to a text file, that will contain a set of sentences. In the Tables 3.5, 3.6, 3.7, 3.8 are presented the number of sentences each subtype has. These subtypes will be our training corpus for the language models. It is important to refer that each sentence just can be assigned to one subtype. It cannot have two subtypes.

Starting with **introduction**, we divide it in the following four subtypes, three regarding the final result and one that fits all the cases:

- **Home team wins:** Appropriate to the matches where the home team wins.
- **Away team wins:** Suitable to the matches where the away team wins.
- **Draw:** Applicable to the matches that end with a draw.
- **Others:** Fits all the cases, not specifying the team that plays home or away but the team that wins or lose.

Subtypes	Nº of sentences
Home team wins	46
Away team wins	26
Draw	24
Others	22
Total	118

Table 3.5: Number of sentences per each subtype of introduction.

Afterwards, **goals** are partitioned in ten subtypes:

- **Increase the advantage:** Appropriate to goals scored by the winning team at the minute of the goal.
- **Decrease the advantage:** Suitable to goals scored by the team that is losing for two or more goals.
- **Win goal:** Applicable to the winning goal of a match.
- **Goal to new advantage:** Fits the cases when the team that scores was earlier in advantage but lost it and, at the minute of the goal, achieves a new advantage in the match.
- **Initial goal:** Appropriate to goals scored in the first fifteen minutes of the match.
- **Match with unique goal:** Suitable to the matches where it was only one goal.
- **First goal:** Applicable to the first goal of a match.

- **Last goal:** Appropriate to the last goal of a match.
- **Draw goal:** Suitable to goals that tie the game.
- **Others:** Fits all the cases, not specifying any of the above characteristics.

Subtypes	Nº of sentences
Increase the advantage	10
Decrease the advantage	6
Win goal	9
Goal to new advantage	6
Initial goal	7
Match with unique goal	12
First goal	16
Last goal	6
Draw goal	14
Others	7
Total	93

Table 3.6: Number of sentences per each subtype of goals.

**Sent-offs** are subdivided in the next three subtypes, two regarding the way the player is expelled and one that do not specify it:

- **Double yellow card:** Applicable when the same player receives two yellow cards followed by a red.
- **Direct red card:** Appropriate when a players gets a straight red card.
- **Others:** Suitable to all the cases since it does not specify how the player is expelled.

Subtypes	Nº of sentences
Double yellow card	13
Direct red card	5
Others	5
Total	23

Table 3.7: Number of sentences per each subtype of sent-offs.

Finally, **conclusion** is divided concerning the final result:

- **Draw:** Applicable when the final result of the match is a draw and each team sums one point to its classification.

- **No draw:** Appropriate when one of the teams win, getting the three points while the losing team stays with the same number of points that had in the beginning of the match. In this case there is no need to refer the home or away team, like in introduction, since the sentences of this category do not mention it.

Subtypes	Nº of sentences
Draw	14
No draw	52
Total	66

Table 3.8: Number of sentences per each subtype of conclusion.

In this process we noticed that some sentences are too specific, which took us to discard this kind of sentences. We qualify a sentence as too specific when the number of possible sentences constituting its subtype is too low. For example, if we have a subtype for when the same player scores three goals in the same game, but we have less than 5 possible sentences to assign to that subtype, we choose to discard those phrases. This is because the larger the training corpus, the better the LM. Even so, some of the training corpus we built are small. This is due to the sparse data. Like explained before, we did a manual analysis of one championship to extract all the data above, which resulted in a small quantity of data and took a large amount of time. Even so we will build Statistical LMs from these corpus and analyze the results.

## 3.4 Language Modeling

In order to build language models and generate sentences from the training corpus, we will use SRILM toolkit.

### 3.4.1 Training Language Models

Having a corpus for each subtype we can now build the Statistical Language Models. SRILM allows to estimate N-gram models, counting all the n-grams in a text file and writing them to a .count file, using *ngram-count*.

---

```
1 ngram-count -text path/to/corpus.txt -order 2 -write path/to/corpus.count
```

---

This command is used to count all the bigrams in a file named corpus.txt and write them to a file denominated corpus.count. Below is presented a small part of the .count file generated with the previous command. The original file is in Portuguese but we translated it to English.

## Statistical Language Modeling

```
1 <away:team> lost, 2
2 <away:team> at 1
3 <away:team> by 24
4 <away:team> lost 5
5 <away:team> was 1
6 <away:team> on 1
```

In order to create a LM we need to add *-lm* option to the command, followed by a path where the LM will be saved. With this option, *ngram-count* will estimate the LM and its probabilities and write them to a .lm file. SRILM estimates a smoothed model by default, so if we want an unsmoothed one we have to add the option *-addsmooth 0*. Smooth is used to avoid probabilities of certain n-grams being equals to zero or one. It is like pretending that a word was seen one more time from what was actually seen. This way the probabilities are more balanced. However, in our case, we want some probabilities to be zero, for example, we want that the sentence always ends with a dot (.) and that is what the LM learns from the training corpus, because it always happen. This way the probability of any other word or symbol ends the sentence is zero. Thus, if we smooth the model every word or symbol can end the sentence, because the model would pretend to have seen another word or symbol ending the sentence one time each, assigning to that n-gram a probability higher than 0.

```
1 ngram-count -text path/to/corpus.txt -order 2 -addsmooth 0 -lm path/to/language/
  model.lm
```

This command is used to build the LM, based on a corpus.txt, and write it to a file named model.lm. In the next listings are presented parts of the model, again translated to English since the original is in Portuguese.

```
1 \data\
2 ngram 1=103
3 ngram 2=213
```

```
1 \2-grams:
2 0 This <weekDay>,
3 0 No terreno
4 -0.1165056 The <home:team>
5 -0.8325089 The <away:team>
6 -1.531479 The <away:team>,
7 -1.531479 The <win:team>
8 -1.531479 The team
9 -1.20412 the <round>
10 -1.20412 by <result:lose:team>
```



```

11 -0.7269987 by <result:lose:team>,
12 -0.90309 by <result:lose:team>.
13 -0.7781513 to <away:team>
14 -0.4771213 to <away:team>.
15 -0.7781513 to beat
16 -0.60206 beat this
17 -0.1249387 beat the
18 -0.30103 beat, at

```

The file will contain a header with the number of n-grams detected, up to a maximum order indicated in the *-order* option, represented in the first list. In the rest of the it is presented a list for each of the indicated n-grams, preceded by the  $\log(\text{base-10})$  of the conditional probability of each list's element. In the second list is presented part of the bigrams list.

Regarding our training corpus we choose to build unsmoothed language models with the maximum order being between 2 and 5. For the small corpus with few sentences we use maximum order 2. Maximum order 5 is more suitable to the larger corpus.

### 3.4.2 Sentences Generation

Having the language models built we just need to generate the sentences. SRILM provides the *ngram* command which allows to generate sentences from a given LM.

```

1 ngram -lm path/to/language/model.lm -gen 1

```

This command generates one sentence, as the given value in the option *-gen* is 1. Below we present an example of a sentence generated from the subtype “home team wins” of introduction.

**Portuguese:** O <equipa:casa> recebeu e venceu o <equipa:fora> por <resultado:equipa:casa> x <resultado:equipa:fora>.

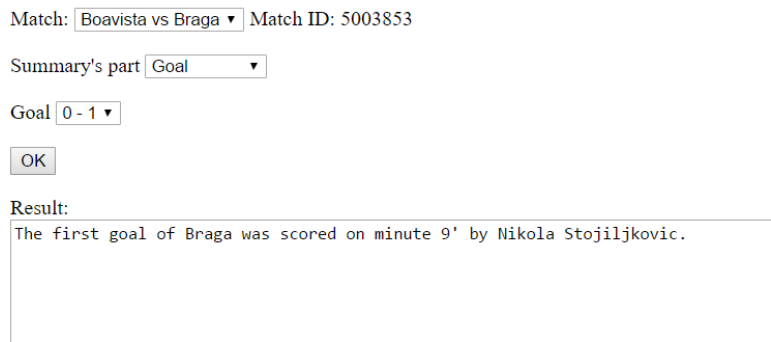
**Translation in English:** <home:team> hosted and beat <away:team> by <result:home:team> x <result:away:team>.

Although the models can generate good sentences, they also generate bad ones, as it will be possible to check in the results. This problem is mainly due to sparsity data. To try to overcome this problem, zerozero.pt provided an additional collection of summaries so we can build a classifier to automatically classify each sentence in one of the four categories that we defined for summaries. This way we will be able to have more data to build models. This process will be explained in Chapter 4.

## 3.5 Implemented System

### 3.5.1 Description of the system

In order to make an usable interface so any user can generate phrases, we developed a system where it is possible to select a match and generate a sentence of a specific category. The process of generating text, from the previously created LMs, with the SRLIM toolkit was integrated in this system. Regarding the system's interface organization, first the user selects the match, then the part of the summary he wants to generate: introduction, goals, sent-offs and conclusion. In goals it is necessary to choose about which goal is intended to produce a sentence. Same for sent-offs where it is necessary to select which one is expected to generate the phrase. When the user clicks the *OK* button, a new sentence is produced from the model. In Figure 3.3 is shown the system's interface, traduced to English.



Match:  Match ID: 5003853

Summary's part

Goal

Result:

The first goal of Braga was scored on minute 9' by Nikola Stojiljkovic.

Figure 3.3: System's interface.

### 3.5.2 Implementation of the system

In the implementation of the system, beyond the LMs and the SRILM toolkit we have access to an API of zerozero.pt that contains information about each match. With this data we can replace the tokens inserted in delexicalization process for real values from a specific match. In the Figure 3.4 is presented a part of the API, about a match from Portuguese League, where Boavista hosted Braga, in the round 21<sup>5</sup>.

When the category of the summary and the event (in the case of being a goal or a sent-off) are chosen, a sentence is generated from the LM suitable for that part and event. The model is chosen based on facts of the selected match. Exemplifying with introduction, the appropriate one is chosen based on the final result. Exists LMs built for each of its subtypes: “home team wins”, “away team wins”, “draw” and “others”. Using the match represented in Figure 3.4, we can see that the final result was 1-1, so the implemented system choose the “draw” model to generate the sentence.

<sup>5</sup><https://www.zerozero.pt/jogo.php?id=5003853>

```

▼ "data": {
  "LEAGUE": "Liga NOS 16/17",
  "STADIUM": null,
  "DATETIME": "2017-02-12 20-15-00",
  "FINALRESULT": "1-1",
  "HOMEGOALS": "1",
  "AWAYGOALS": "1",
  "POINTSHOME": "26",
  "CLASSIFHOME_PRE": "11",
  "CLASSIFHOME_POS": "11",
  "POINTSAWAY": "38",
  "CLASSIFAWAY_PRE": "4",
  "CLASSIFAWAY_POS": "4",
  "JORNADA": "21",
  ▼ "MATCHREPORT": {
    ▼ "edicao": {
      "id": "98399",
      "com_cartoes_azuis": "0",
      "fk_epoca": "146",
      "descr_epoca": "2016/17",
      "tipo_fichas_jogos": "0",
      "stats_nivel": "30"
    },
    ▼ "jogo": {
      "id": "5003853",
      "fk_equipa_casa": "5",
      "fk_equipa_fora": "15",
      "abrev_equipa_casa": "Boavista",
      "abrev_equipa_fora": "Braga",
      "fk_estado": "1",
      "te_modalidade_casa": "1",
      "te_modalidade_fora": "1",
      "te_modalidade": "1"
    }
  }
}

```

Figure 3.4: Part of the match information, provided by zerozero.pt API, in JSON.

**Generated sentence:** The team of <coach:away:team> did not go beyond a tie by <result:home:team> x <result:away:team> against <home:team>.

The sentence is delexicalized, so by accessing the API we are able to get the respective values. The final sentence, after the tokens replace, is presented next.

**Final sentence:** The team of Jorge Simão did not go beyond a tie by 1 x 1 against Boavista.

The results of this process will be analyzed and discussed in Chapter 5.



## Chapter 4

# Automatic Classification of Summaries

### 4.1 Classifier

To try to overcome the data sparsity in the first created LMs, we asked `zerozero.pt` to provide an additional collection of summaries. These summaries can be automatically classified using a classifier. This classifier will be built using the `scikit-learn` toolkit and will assign each sentence to one of the four categories that we defined for summaries.

#### 4.1.1 Implementation and Classification

`Scikit-learn` is described as a Python module for machine learning [PVG<sup>+</sup>11]. It can be used in classification, regression, clustering, dimensionality reduction, model selection and preprocessing. We use it in classification, since we want to identify which category each sentence belongs to. In the implementation of the classifier we use as train data all the sentences which constitute the summaries that we manually extracted from the Italian championship, before they were delexicalized, what gives a total of 486. Each sentence was assigned to a category and is in the following format:

---

```
1 category | sentence
```

---

The test data will be the new summaries provided, not assigned to any category yet. In our dataset we have the attributes, also known as features that correspond to the sentences and the categories are the responses, also known as labels or output. First we represent each category in numerical data:

---

```
1 introduction: 0
2 goals: 1
3 sent-offs: 2
4 conclusion: 3
```

---

Then we use `CountVectorizer`<sup>1</sup> in features, so the text is converted into a matrix of word counts, since most of the algorithms expect numerical features vectors with invariable size. The vector learns the vocabulary of the training data and transform it into a document-term matrix. This is a sparse matrix with the shape of *rows x columns* where rows represent the number of sentences and columns represent all the different words that composes the vocabulary. It saves the words counts of each sentence. This strategy is called bag of words, where each sentence is described by word occurrences, disregarding the words position in the document. It also lowercases every word and ignores symbols and stop words.

	about	here	is	live	this	you
Doc 0	0	1	0	1	0	1
Doc 1	1	0	1	0	1	1

Table 4.1: Document-term matrix example.

The test data is also transformed in a document-term matrix, using the vocabulary of the training data and ignoring words that were not there. To build the model we use the multinomial Naive Bayes classifier<sup>2</sup> since it is appropriate for classification with discrete features as word counts for classification of text and it requires integer feature counts. The training data is used to train the model and then it make class predictions for the test data, assigning each sentence to its corresponding category.

To calculate the accuracy score of the classifier we divide the training data in two parts: 75% remains in the training data and 25% is assigned to a test dataset, randomly. This way both train and test sentences are labeled with summaries categories. The process is the same that we explained above but we use this dataset. After predicting the model built for the test data we compare the initial test data (25% of training data) and the prediction of the model. We have a result of 95% of accuracy, so we consider the classifier as good and reliable.

## 4.2 Data preparation and Modulation

The classifier divided the test data like is shown in the Table 4.2. We decided to just use one of the categories dataset due to the lack of time. The one that we think is more appropriate is introduction because it is possible to delexicalize using the API data of each match.

<sup>1</sup>[http://scikit-learn.org/stable/modules/generated/sklearn.feature\\_extraction.text.CountVectorizer.html](http://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.CountVectorizer.html)

<sup>2</sup>[http://scikit-learn.org/stable/modules/generated/sklearn.naive\\_bayes.MultinomialNB.html](http://scikit-learn.org/stable/modules/generated/sklearn.naive_bayes.MultinomialNB.html)

## Automatic Classification of Summaries

Categories	Nº of sentences
Introduction	1428
Goals	4315
Sent-offs	25
Conclusion	979
Total	6747

Table 4.2: Number of sentences per category.

All the sentences have a match ID associated, so we can access the correct one in the API. For each sentence, we automatically replace all the words that were equal in both places, or very similar, for the respective token. Then we create a criteria to select the more suitable sentences. We mark every capital word, except the initial word of a sentence, and count them to see how many entities are still in the sentence, since the entities are usually data that should be delexicalized. Having this count we only select the delexicalized phrases that have two or less remaining entities and three or more tokens, to assure that the selected ones have good quality. These are next divided in the subtypes previously presented in Table 3.5 except “others”. Through the API we know the final result of the game in order to designate each sentence to the correct subtype. After this, it is necessary to manually check each one to delete the entities that are not present in the API, and the words that link them to the sentences, so they remain well constructed, and the entities are not added to the LM’s vocabulary. The final step is to merge the sentences of this training data with the one used in Chapter 3. The sentences are divided in subtypes as shown in Table 4.3.

Subtypes	Nº of sentences
Home team wins	181
Away team wins	97
Draw	87
Total	365

Table 4.3: Number of sentences per introduction’s subtype.

Having this new and larger training corpus we repeat the modelation process presented in Chapter 3 and generate new sentences. The results of this process will also be analyzed and

discussed in Chapter [5](#).



## Chapter 5

# Evaluation

In this chapter is presented the methodology used to evaluate the implemented system and the used LMs to generate sentences, and the results analysis and discussion.

### 5.1 Methodology

Evaluate the created system and the models built is a very important task so it can be possible to analyze the produced text and discuss the obtained results. As stated in Section 2.5, there are multiple ways to evaluate an NLG system. We intend to evaluate the quality of the produced text using manual evaluation. We chose not to use any automatic evaluation, like BLEU metric, which compares the generated text from the system with a set of texts authored by humans, word by word, to check the similarity. We found this not very useful since comparing the produced sentences with human-authored ones might be inaccurate since a writer uses background knowledge and his own opinions, that is not possible to access in the API's data, to complement the text. Our goal is to generate sentences with all the important information, well-written and easy to understand.

We will evaluate sentences generated by two different corpus:

1. Constituted by sentences built after analyzing all the news written about the matches from the season 2015/2016 of the Italian championship, Serie A, using the process demonstrated in Chapter 3.
2. Composed by sentences built after the analysis of all the news written about the matches from the season 2015/2016 of the Italian championship, Serie A, plus a large set of news from where we were able to extract the ones that fits the corpus, with the process explained in Chapter 4.

We generated sentences using both of the corpus and we will evaluate them. The necessity to create a larger corpus emerged when we noticed that some corpus of the first set, after being divided in categories and then in subtypes, became very small, a few composed by less than 10 sentences. The second corpus just includes the category introduction.

## Evaluation

Corpus 1	Corpus 2
Includes the following categories: Introduction, Goals, Sent-offs and Conclusion.	Only includes the category Introduction.
Composed by all the subtypes of the referred categories.	Composed by all the subtypes of Introduction, except “Others”, due to reasons already clarified.
Each corpus file corresponds to a subtype.	Each corpus file corresponds to a subtype.

Table 5.1: Corpus characterization.

Two questionnaires were written to assess the quality of the sentences generated, one per each corpus, and we will make a comparison when it is pertinent, since one is about all of the categories and the other only regards one of them. The first includes sentences of the four categories of summaries and the system that generated its sentences use the first corpus. The second only has introduction related phrases and the corpus used is the second one. In both questionnaires is presented the gamesheet of the match in question. The intelligibility and the completeness of each sentence is evaluated by making two questions:

**In your opinion, and superficially analyzing the gamesheet, the generated sentences are intelligible?**

- **Min (1/5):** The sentence is imperceptible.
- **Max (5/5):** The sentence is lexically and grammatically correct.

**Are the sentences complete, taking into account the summary’s category where they are included?**

- **Min(1/5):** It is missing key information, what makes the sentence very incomplete.
- **Max(5/5):** All key information is presented, so the sentence is appropriate.

At the beginning of each inquiry is explained its **structure**. In the first inquiry, three sentences from each category are evaluated, giving a total of 15. A match is presented, followed by sentences of the category “introduction”, then two “goals” of different subtypes and finally the “conclusion”.

## Evaluation

After that, is presented another match so we can also include the category “sent-offs”, since the first match did not have this one. In Table 5.2 is presented the subtype/s of each set of sentences.

Match	Category	Subtype
Feirense vs Rio Ave	Introduction	Home team wins
	Goals	First goal / Initial goal
	Goals	Decrease the advantage
	Conclusion	No draw
Belenenses vs Braga	Sent-offs	Double yellow card / Others

Table 5.2: Subtype of each set of sentences on questionnaire 1.

The second questionnaire just includes the category introduction, so we used three different games and generated four sentences for each one, having a total of 12. The subtype of each set of sentences is presented in Table 5.3. As previously explained, all of these sentences are evaluated by the users answers to the two questions presented in the list above. A part of the inquiry structure can be seen in Figure 5.1.

Match	Category	Subtype
Chaves vs Arouca	Introduction	Home team wins
Tondela vs Feirense	Introduction	Away team wins
Boavista vs Braga	Introduction	Draw

Table 5.3: Subtype of each set of sentences on questionnaire 2.

## Estrutura

Na geração de texto cada notícia acerca de um jogo foi dividida em 4 partes: Introdução, Golos, Expulsões (se existirem) e Conclusão.  
Foram geradas 3 frases para cada parte da notícia de modo a avaliar a qualidade do sistema.

## Feirense 2 - 1 Rio Ave

Ficha de jogo: <http://www.zerozero.pt/match.php?id=5003844>

Na sua opinião, e fazendo uma análise superficial à ficha de jogo, as frases geradas são inteligíveis (de fácil compreensão)?

Estão completas tendo em conta a parte da notícia em que se incluem?

## Introdução - Inteligibilidade

O Feirense recebeu e venceu o Rio Ave por 2 x 1. \*

	1	2	3	4	5	
A frase é imperceptível.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	A frase é lexicalmente e gramaticalmente correta.

O Feirense, equipa orientada por 2 x 1, em jogo a contar para a 1, esta segunda-feira. \*

	1	2	3	4	5	
A frase é imperceptível.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	A frase é lexicalmente e gramaticalmente correta.

Figure 5.1: Image of the questionnaire's structure.

## 5.2 Results and Discussion

### 5.2.1 Evaluation of the implemented system's output

In the evaluation of the final sentences generated by the implemented system we did two inquiries, as previously stated. The first one was answered by fifty-one evaluators, previously not involved in the project. The second survey was responded by fifty-two evaluators, all different from the first one. The average classification per sentence is represented in Figures 5.2 and 5.5, for each one of the questionnaires.

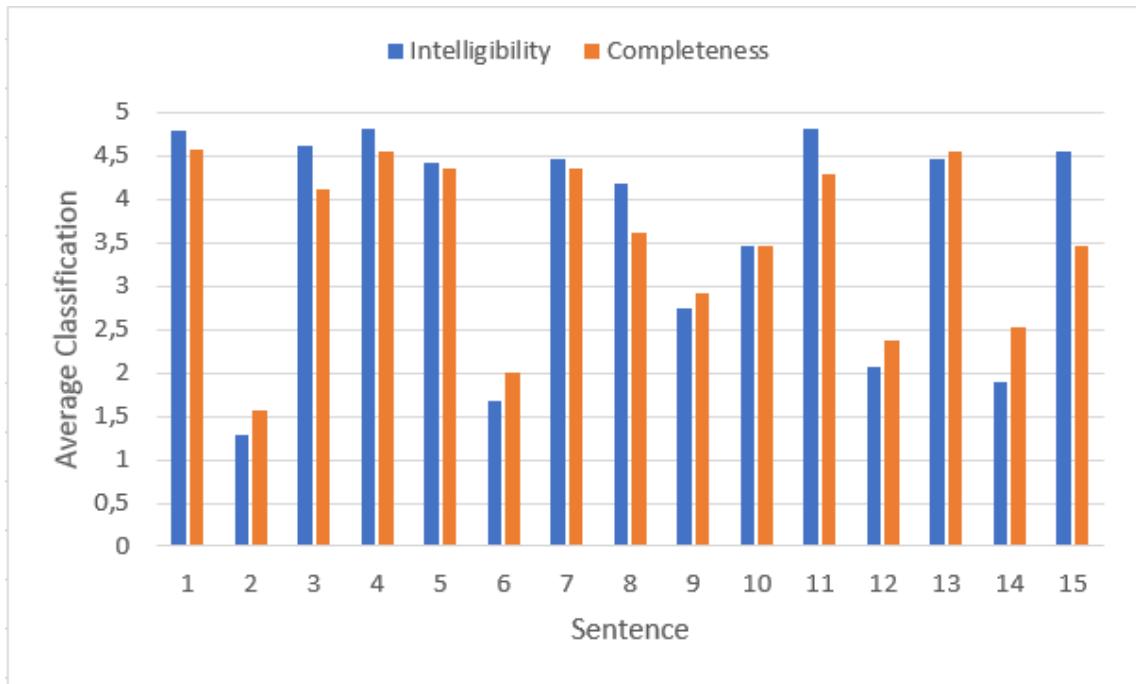


Figure 5.2: Average classification per sentence on the first questionnaire.

The average intelligibility score, on the first questionnaire, is 3.61 out of 5. The completeness average classification is 3.52 out of 5. The difference between these values is 0.09 so we conclude that the level of intelligibility and completeness is very similar, as the system can generate the same amount of complete sentences and lexically and grammatically correct ones. We consider classifications below 3 as negative since the scale goes from 1 to 5, being 3 the intermediate value. Out of the fifteen sentences, the number of sentences which scored below 3 are four, in intelligibility, and three in completeness. On the other hand, the number of sentences that scored above or equal to 4.5 are five in intelligibility and three in completeness. It is possible to conclude that the system can both generate sentences of very good quality and very poor ones.

In Figure 5.3 is represented a stacked bar chart that shows the distribution of the sentences classification, regarding intelligibility, by sentence. Stacked bar charts used with a 100% scale, in this case and in the following with the same type of chart, are useful to understand the percentage of each possible classification for each sentence. This will help us to simultaneously compare the

## Evaluation

percentages and notice sharp changes between sentences. It is possible to verify that sentences 2, 6, 9, 12 and 13 have an high percentage (more than 50%) of negative values (below 3, the intermediate value). All the other sentences have a big percentage of positive values, which indicate that those sentence scores a high value in intelligibility. We also assess that the most common classification in the sentences overall is 5, with a percentage of 44.6%. To conclude there was a total of 61.7% of positive values, 9,5% of intermediate ones and 28.8% of negative values.

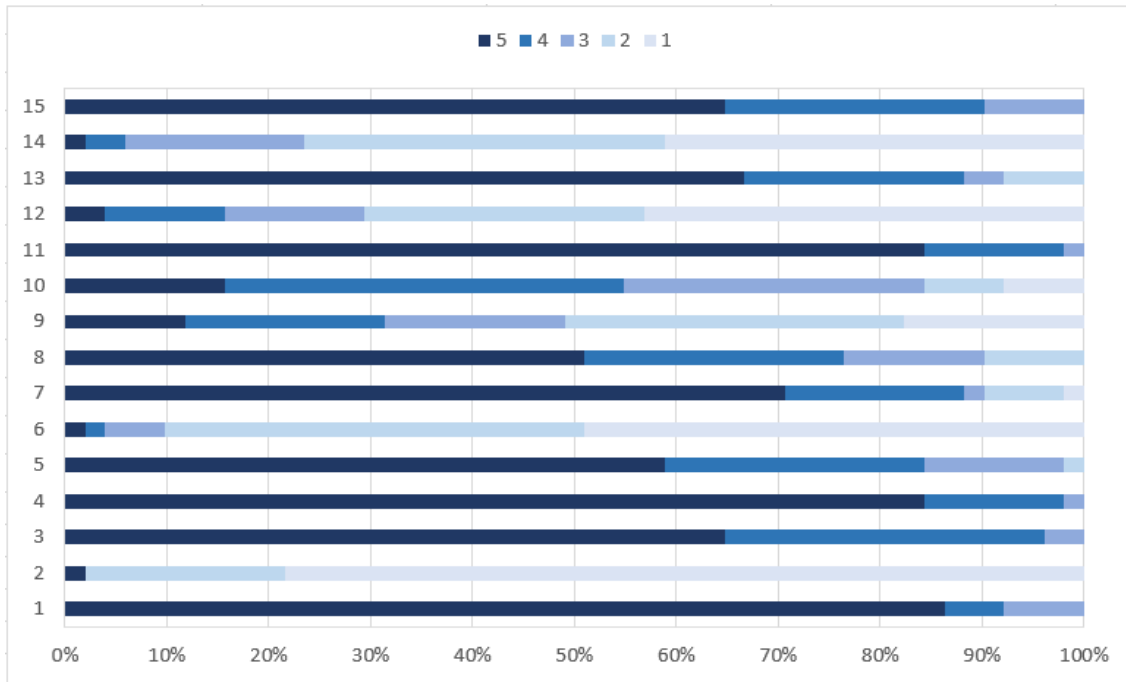


Figure 5.3: Distribution of the sentences classification on the first questionnaire, for intelligibility.

Figure 5.4 shows a stacked bar chart that presents the distribution of the sentences classification, regarding completeness, per sentence. It is visible that sentences 2, 6, 12, 14 have an high percentage of negative values. Sentences 9 and 15 scored an intermediate value, not having neither a high percentage of positive values nor negative values. All the other sentences have a big percentage of positive values which indicate that those sentence scores a high value in completeness. In this one, comparing with the intelligibility one, there is a fewer percentage of 5's in the overall, corresponding to 35.1%. On the other hand it is possible to understand that the quantity of 4's and 3's, in overall, is higher than in intelligibility. In completeness, evaluators opted for not classifying the sentences with the higher value, using 4's and 3's instead. The percentage of negative values is 27.6%, a very similar value when comparing with intelligibility distribution. Positive values scored a total of 56.8% and the intermediate one 15.6%.

## Evaluation

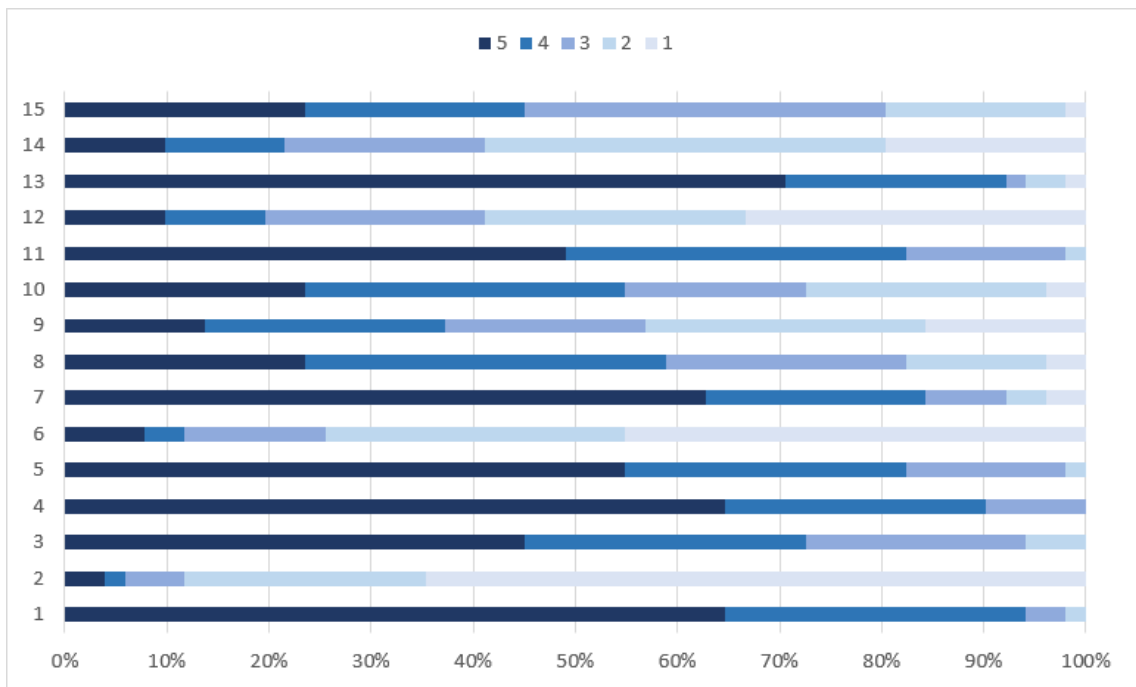


Figure 5.4: Distribution of the sentences classification on the first questionnaire, for completeness.



Figure 5.5: Average classification per sentence on the second questionnaire.

On the second inquiry, the intelligibility average score is 3.6 out of 5. The completeness average classification is 3.32 out of 5. In this case, the difference was larger than in the first questionnaire, with a value of 0.28. We believe this is due to the fact that in this survey we are just evaluating sentences from one category, introduction, which requires a large amount of data from the API. In

## Evaluation

category goals or sent-offs it is easier to have more complete phrases because it requires less input data. The number of sentences which scored below 3 in intelligibility are four and in completeness are three, out of twelve sentences. On the other hand, the number of sentences that scored above or equal to 4.5 are four in intelligibility and two in completeness.

In order to analyze the distribution of the sentences classification, regarding intelligibility per sentence, in this second questionnaire we also present a stacked bar chart, in Figure 5.6. The sentences with a percentage of negative values higher than 50% are 3, 6, 10 and 12. All the other eight sentences scored an high percentage of positive classifications, indicating that those sentences are considered intelligible. In overall, the percentage of positive classifications was 61%, the intermediate value scored 11.4% and the negative ones have a score of 27.6%.

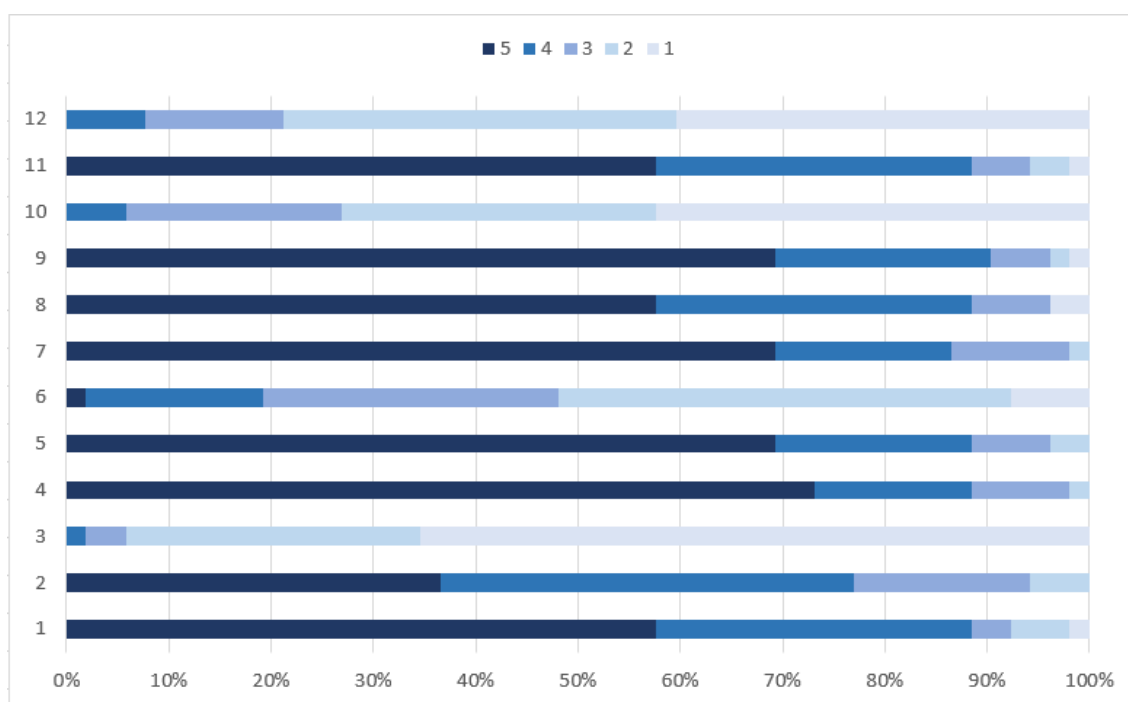


Figure 5.6: Distribution of the sentences classification on the second questionnaire, for intelligibility.

Figure 5.7 presents another stacked bar chart which shows the distribution of the sentences classification, regarding completeness, per sentence. Sentences 3, 10 and 12 scored a high percentage of negative values, so they are not considered complete. Regarding high percentages of positive values, sentences 1, 2, 7, 8, 9 and 11 stood out. The ones that scored an intermediate value were sentences 4, 5 and 6. We notice that in this distribution the amount of 5's reduced drastically and the quantity of 4's and 3's increased, comparing with intelligibility. Evaluators opted for not classifying the sentences with the higher value, using 4's and 3's instead, just like happened in the first questionnaire, in completeness. The overall percentage of positive values was 50.2%, intermediate value percentage was 22.4% and negative values scored a percentage of 27.4%.

Next we are going to evaluate the system with both corpus, regarding the number of words of each sentence and the impact of categories and subtypes in both intelligibility and completeness.



## Evaluation

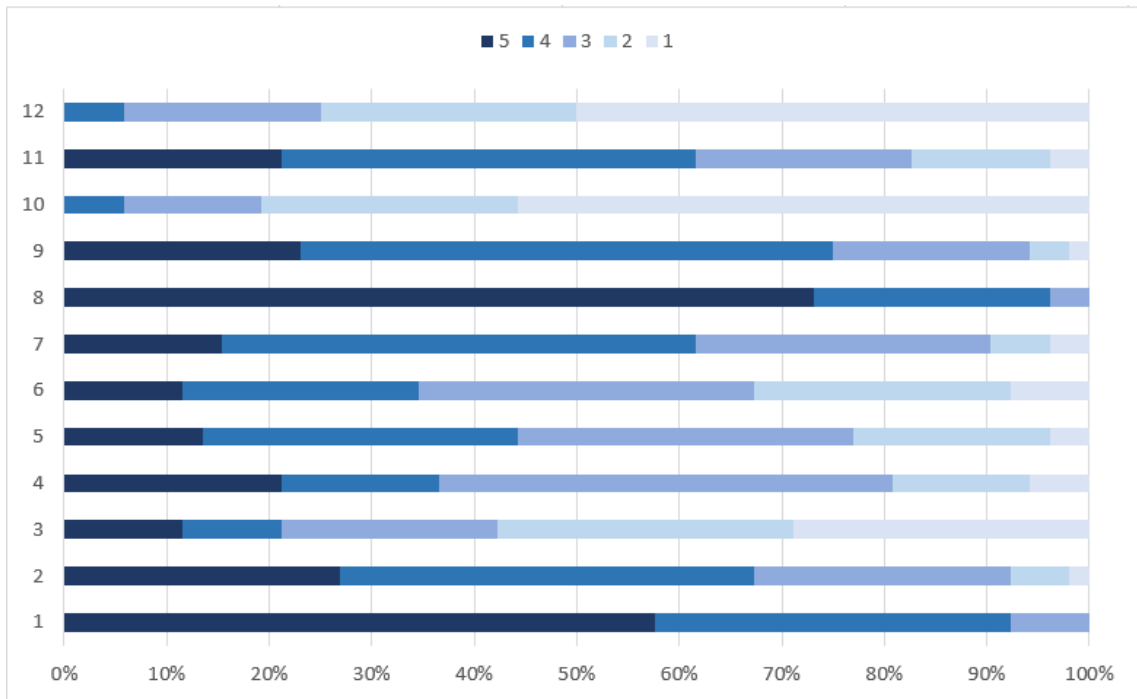


Figure 5.7: Distribution of the sentences classification on the second questionnaire, for completeness.

We divided each set of sentences in three groups in order to evaluate if the increase of the number of words per sentence is connected with the variation of intelligibility or completeness. Each part has similar length. In Figure 5.8 we can see how words per sentence influence both intelligibility and completeness on the sentences that compose questionnaire one. With the increase of the number of words we notice a big decrease of the intelligibility score. From the first group ([0-9]) to the second ([9-18]) the average classification passed from 4.49 to 3.78, which is a significant loss. From the second to the third group([18,25]) the decrease was bigger, going from 3.78 to 2.27. We believe this happens because we are working with Statistical LMs. Since the next word in a sentence is chosen based on probabilities, the more words it generates the higher the chance of choosing a word that does not fit the context of the previous ones, decreasing the intelligibility. As for the completeness there is also a decrease, but this is smaller. From group one to group two the loss is from 4.07 to 3.67 and from the second to the third group the decrease is from 3.67 to 2.59, a larger decrease than the previous one. We found the decrease of completeness odd because usually the larger the sentence more information it can have. However, as we are using Statistical LMs we have the same problem stated before. The next word can be a token and if the same tokens keep being chosen by the LM the sentence will be repeating information.

In Figure 5.9 we can check the influence that words per sentence has in intelligibility and completeness, in sentences generated using the second corpus has less variation than the ones using the first corpus. As for the completeness the average classification did not change enough to say that words per sentence influenced it. We conclude that all the generated sentences have the same level of completeness. Regarding intelligibility, from the first group([0,11]) to the second ([11,18])

## Evaluation



Figure 5.8: Intelligibility and completeness average classification regarding the number of words per sentence on the first questionnaire.

the variation is very small, passing from 3.93 to 3.80. In the transition from the second to the third group ([18,25]) the variation is higher, going from 3.80 to 2.81. Since the category of the corpus used to generate the sentences we are analyzing, on the questionnaire two, is introduction and the length of them is larger than the ones used to generate the sentences on the first questionnaire, we conclude these corpus are more suitable to sentences that have a length until eighteen words. Higher than data will probably have worst results.

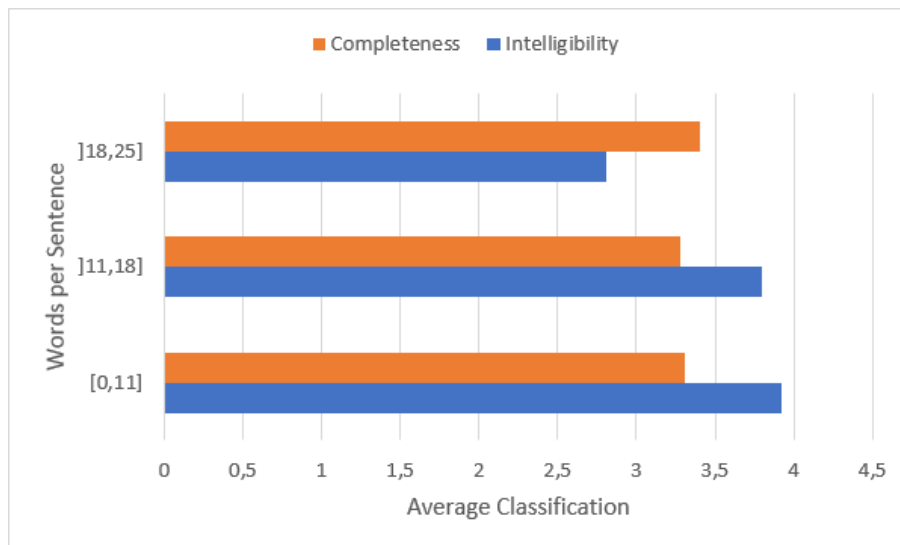


Figure 5.9: Intelligibility and completeness average classification regarding the number of words per sentence on the second questionnaire.

Now we are going to analyze them based on the categories and subtypes. Using the Figure 5.10, regarding questionnaire one, it is possible to compare the intelligibility and completeness

## Evaluation

between each one of the categories. Concerning completeness, “goals” scored the higher value, followed by “sent-offs”, “introduction” and “conclusion” in last. As stated before its easier for the categories “goals” and “sent-offs” to achieve higher results in completeness since they need less information when comparing to “introduction” and “conclusion”. As for intelligibility the order is the same. We conclude that this can be due to the fact of sentences about “goals” and “sent-offs” are more objective and, having less information the sentence can be smaller what helps the sentence to be easier to read, comparing to “introduction” and “conclusion”.

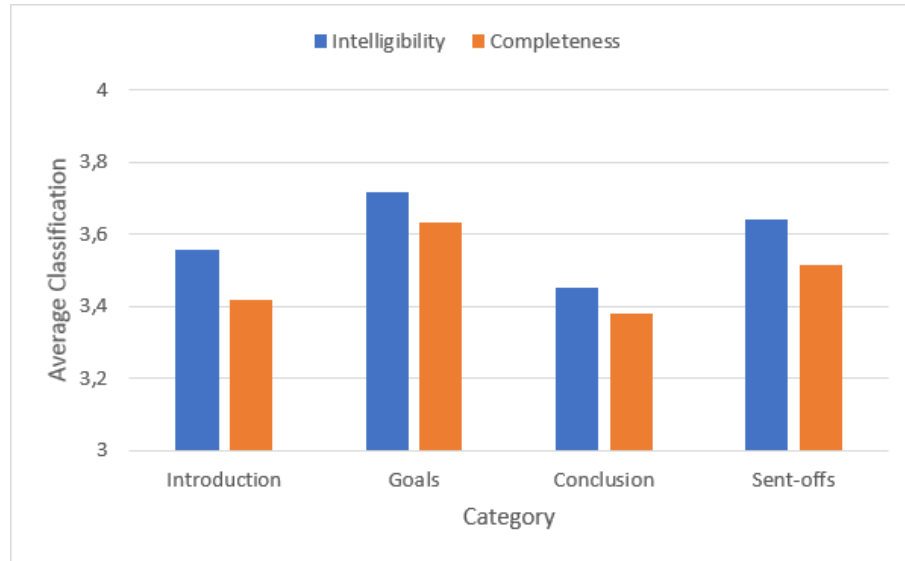


Figure 5.10: Intelligibility and completeness average classification regarding the sentence’s category on the first questionnaire.

Questionnaire two is only composed by one category (Figure 5.11), as stated before and the average scores were also previously discussed. It is unfair to compare the scores obtained in “introduction” from questionnaire one and two since the first is only composed by three sentences and the second is composed by twelve.

On the second survey we use the three subtypes of “introduction”: “home team wins”, “away team wins” and “draw”. In Figure 5.12 is presented how intelligibility and completeness score in each one. Normally the order of these scores would follow the size of the corpus, being the “home team wins” the bigger one, followed by “away team wins” and “draw” in last, but that did not happen. The “draw” subtype scores, without surprise, the worst average classification, but the “away team wins” subtype has a better average classification than “home team wins”, even having a smaller corpus. We believe this happens because the sentences that compose “away team wins” corpus has less variation when comparing with “home team wins” corpus. This can make them score higher, because with less variation the options of the next words, in the LM, would be fewer, decreasing the chance of choosing a word that does not fit the context of the previous ones.

## Evaluation



Figure 5.11: Intelligibility and completeness average classification regarding the sentence's category on the second questionnaire.

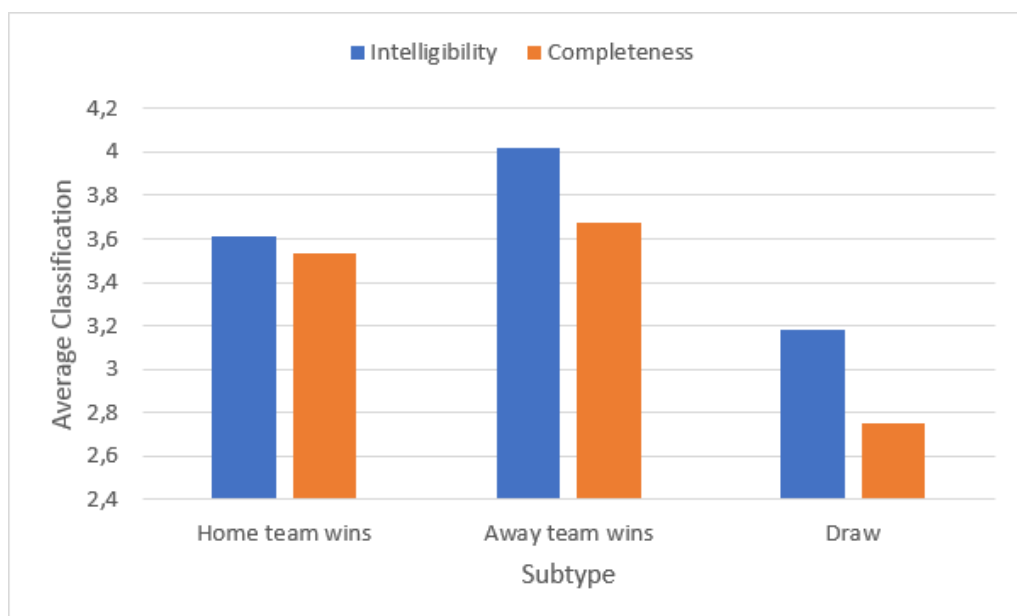


Figure 5.12: Intelligibility and completeness average classification regarding the sentence's subtype on the second questionnaire.

## Chapter 6

# Conclusions and Future Work

### 6.1 Summary

The main goal of this dissertation was to use Statistical LMs to generate sentences that can be part of a summary. Nowadays there is a need to speed up the news production process because there are a large amount of matches that people wants to know about. Zerozero.pt, saves data from more than 6000 matches per week but only has the means to produce news for an average of 100 of those matches. With this large amount of structured data it is possible to generate news using NLG techniques. This would help the journalist writing their pieces in less time, since that would summarize the data in text instantly.

In order to achieve the objectives we started by reviewing the state of the art regarding Natural Language Generation. We decided to use a statistical approach, Statistical LMs, which are based in probabilities extraction from a corpus, using N-grams. Zerozero.pt provided all the data that we needed to build the corpus. The extraction of sentences was done manually, with us analyzing each summary from the season 2015/2016 of the Italian championship, Serie A. After this, we manually delexicalized all the extracted summaries, using tokens, in order to generalize all the text so it is prepared to fit any possible match. In all the summaries we found a pattern: first the journalist introduces the game, then he talks about the goals and sent-offs if there were any and finalize it with a conclusion. Due to this reason we divided each summary in four categories: introduction, goals, sent-offs and conclusion. In each category the text was divided by sentences. After having all the sentences categorized we needed to divide each one of the categories in subtypes because the sentences can be specific to a certain part of the game or a certain result. For example we have a subtype “home team wins” for introduction, only suitable to the sentences that indicates that fact.

With the help of the SRILM toolkit we trained language models using the corpus built before and generated delexicalized sentences. In order to any user be able to generate sentences about a specific match we implemented a system that automatically identifies the subtype of the selected category and replaces the tokens with match information from the API, generating the final sentence, without tokens. These first models created allows to both generate good sentences and poor quality ones as we can assess in Chapter 5. We believed this was due to sparsity data. Some of the

corpus was composed by less than ten sentences, so the need of creating larger corpus emerged. We decided to create a classifier, using the scikit-learn toolkit. As training data we used the summaries used to build the first corpus. We also had access to a larger amount of news provided by zerozero.pt so we could classify each sentence in one of the categories defined. These news were used as test data. Due to the lack of time to finish this work we only used the sentences classified as “introduction”. This way we increased the “introduction” collection three times. In order to generate sentences using this corpus, we trained language models and used the implemented system to automatically generate the final sentences.

We used manual evaluation to assess the quality of the system and the generated sentences. Two questionnaires were made: the first regarding the sentences generated using the first corpus and the second one including phrases generated from the second corpus. In the first one, intelligibility scored an average of 3.61, out of 5 and completeness scored 3.52, out of 5. In the second one, intelligibility had an average score of 3.6, out of 5 and the completeness scored 3.32, out of 5. We noticed that there was not a big change between the results of the two questionnaires but we have to remember that the first one is composed by 15 sentences that include all the four categories and the second one has 12 sentences but all from the same category. Each category has different characteristics so the comparison between questionnaires is not fair. We assessed this by comparing the intelligibility and completeness by category, concluding that with “goals” and “sent-offs” is easy to generate phrases of higher quality. However we expected to notice a higher score in the second questionnaire’s results. This may have happened for some reasons: despite the increase of the corpus size perhaps this increase was not enough and a bigger corpus is necessary to have better results; maybe it was necessary to evaluate a larger number of sentences in order to understand the quality of each subtype’s collection of sentences. Even so we consider the results positive and we think we made a good contribution to this promising area.

## 6.2 Future Work

In the final of this dissertation we state that a part of the generated sentences have good quality but the system can also generated very poor ones, so we conclude that it is possible to generate even better sentences. There are multiple options to improve the system’s output quality. The first one is getting a very large number of sentences for each corpus. However this one requires even more manual work and many temporal resources in order to just have sentences of good quality on the corpus, all delexicalized and well divided by category and subtype. Another possibility is to use another approach, using learning algorithms like deep-learning or neural networks.

# Appendix A

## Questionnaire one

In this appendix are presented images from the questionnaire one.

**Geração Automática de Notícias de Desporto**

Este inquérito destina-se à avaliação de frases geradas automaticamente por um sistema desenvolvido no âmbito da unidade curricular Dissertação, do Mestrado Integrado em Engenharia Informática e Computação.

Agradeço, desde já, a sua participação. Para alguma questão contacte:  
[joao.ricardo.soares@fe.up.pt](mailto:joao.ricardo.soares@fe.up.pt)

\*Required

**Estrutura**

Na geração de texto cada notícia acerca de um jogo foi dividida em 4 partes: Introdução, Golos, Expulsões (se existirem) e Conclusão.  
Foram geradas 3 frases para cada parte da notícia de modo a avaliar a qualidade do sistema.

**Feirense 2 - 1 Rio Ave**

Ficha de jogo: <http://www.zerozero.pt/match.php?id=5003844>

Na sua opinião, e fazendo uma análise superficial à ficha de jogo, as frases geradas são inteligíveis (de fácil compreensão)?

Estão completas tendo em conta a parte da notícia em que se incluem?

**Introdução - Inteligibilidade**

O Feirense recebeu e venceu o Rio Ave por 2 x 1. \*

	1	2	3	4	5
A frase é imperceptível.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
A frase é lexicalmente e gramaticalmente correta.					

Figure A.1: Part 1 of questionnaire one.

## Questionnaire one

O Feirense, equipa orientada por 2 x 1, em jogo a contar para a 1  
, esta segunda-feira. \*

	1	2	3	4	5	
A frase é imperceptível.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	A frase é lexicalmente e gramaticalmente correta.

Feirense e Rio Ave defrontaram-se, esta segunda-feira, em encontro da 20.ª jornada da Liga Portuguesa. \*

	1	2	3	4	5	
A frase é imperceptível.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	A frase é lexicalmente e gramaticalmente correta.

### Introdução - Completitude

O Feirense recebeu e venceu o Rio Ave por 2 x 1. \*

	1	2	3	4	5	
Falta informação chave, o que torna a frase muito incompleta.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Toda a informação chave é apresentada, sendo assim a frase é adequada.

Figure A.2: Part 2 of questionnaire one.



## Questionnaire one

O Feirense, equipa orientada por 2 x 1, em jogo a contar para a 1, esta segunda-feira. \*

	1	2	3	4	5	
Falta informação chave, o que torna a frase muito incompleta.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Toda a informação chave é apresentada, sendo assim a frase é adequada.

Feirense e Rio Ave defrontaram-se, esta segunda-feira, em encontro da 20.<sup>a</sup> jornada da Liga Portuguesa. \*

	1	2	3	4	5	
Falta informação chave, o que torna a frase muito incompleta.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Toda a informação chave é apresentada, sendo assim a frase é adequada.

[NEXT](#)

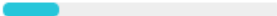
 Page 1 of 5

Figure A.3: Part 3 of questionnaire one.

## Questionnaire one

**Feirense 2 - 1 Rio Ave**

Ficha de jogo: <http://www.zerozero.pt/match.php?id=5003844>

**Golo (1-0) - Inteligibilidade**

Higor Platiny inaugurou o marcador aos 6'. \*

	1	2	3	4	5	
A frase é imperceptível.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	A frase é lexicalmente e gramaticalmente correta.

Higor Platiny abriu o marcador à passagem do minuto 6'. \*

	1	2	3	4	5	
A frase é imperceptível.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	A frase é lexicalmente e gramaticalmente correta.

O conjunto orientado por Nuno Manta, que jogou na condição de equipa casa, a turma orientada por Nuno Manta, que jogou na frente do minuto 6'. \*

	1	2	3	4	5	
A frase é imperceptível.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	A frase é lexicalmente e gramaticalmente correta.

Figure A.4: Part 4 of questionnaire one.

## Questionnaire one

**Golo (1-0) - Completitude**

Higor Platiny inaugurou o marcador aos 6'. \*

	1	2	3	4	5	
Falta informação chave, o que torna a frase muito incompleta.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Toda a informação chave é apresentada, sendo assim a frase é adequada.

Higor Platiny abriu o marcador à passagem do minuto 6'. \*

	1	2	3	4	5	
Falta informação chave, o que torna a frase muito incompleta.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Toda a informação chave é apresentada, sendo assim a frase é adequada.

O conjunto orientado por Nuno Manta, que jogou na condição de equipa casa, a turma orientada por Nuno Manta, que jogou na frente do minuto 6'. \*

	1	2	3	4	5	
Falta informação chave, o que torna a frase muito incompleta.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Toda a informação chave é apresentada, sendo assim a frase é adequada.

BACK

NEXT

Page 2 of 5

Figure A.5: Part 5 of questionnaire one.

## Questionnaire one

**Feirense 2 - 1 Rio Ave**

Ficha de jogo: <http://www.zerozero.pt/match.php?id=5003844>

**Golo (2-1) - Inteligibilidade**

O Rio Ave ainda reduziu, aos 90', por Gonçalo Paciência. \*

	1	2	3	4	5	
A frase é imperceptível.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	A frase é lexicalmente e gramaticalmente correta.

Gonçalo Paciência reduziu, e reabriu a discussão pelo resultado. \*

	1	2	3	4	5	
A frase é imperceptível.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	A frase é lexicalmente e gramaticalmente correta.

Gonçalo Paciência reduziu para o Rio Ave aos 90', por Gonçalo Paciência. \*

	1	2	3	4	5	
A frase é imperceptível.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	A frase é lexicalmente e gramaticalmente correta.

Figure A.6: Part 6 of questionnaire one.

## Questionnaire one

### Golo (2-1) - Completitude

O Rio Ave ainda reduziu, aos 90', por Gonçalo Paciência. \*

	1	2	3	4	5	
Falta informação chave, o que torna a frase muito incompleta.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Toda a informação chave é apresentada, sendo assim a frase é adequada.

Gonçalo Paciência reduziu, e reabriu a discussão pelo resultado. \*

	1	2	3	4	5	
Falta informação chave, o que torna a frase muito incompleta.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Toda a informação chave é apresentada, sendo assim a frase é adequada.

Gonçalo Paciência reduziu para o Rio Ave aos 90', por Gonçalo Paciência. \*

	1	2	3	4	5	
Falta informação chave, o que torna a frase muito incompleta.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Toda a informação chave é apresentada, sendo assim a frase é adequada.

BACK

NEXT

Page 3 of 5

Figure A.7: Part 7 of questionnaire one.

**Feirense 2 - 1 Rio Ave**

Ficha de jogo: <http://www.zerozero.pt/match.php?id=5003844>

**Conclusão - Inteligibilidade**

O Feirense no 13º posto com 22 pontos. Já o Rio Ave continua com 27, na classificação da Liga Portuguesa. \*

	1	2	3	4	5	
A frase é imperceptível.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	A frase é lexicalmente e gramaticalmente correta.

Com este triunfo, o Feirense segue na 13ª posição com 22 pontos. \*

	1	2	3	4	5	
A frase é imperceptível.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	A frase é lexicalmente e gramaticalmente correta.

Com este resultado, o Rio Ave é 9ª posição da Liga Portuguesa com 27 pontos e está no 9º lugar com 22 pontos. \*

	1	2	3	4	5	
A frase é imperceptível.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	A frase é lexicalmente e gramaticalmente correta.

Figure A.8: Part 8 of questionnaire one.

**Conclusão - Completitude**

O Feirense no 13º posto com 22 pontos. Já o Rio Ave continua com 27, na classificação da Liga Portuguesa. \*

	1	2	3	4	5	
Falta informação chave, o que torna a frase muito incompleta.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Toda a informação chave é apresentada, sendo assim a frase é adequada.

Com este triunfo, o Feirense segue na 13ª posição com 22 pontos. \*

	1	2	3	4	5	
Falta informação chave, o que torna a frase muito incompleta.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Toda a informação chave é apresentada, sendo assim a frase é adequada.

Com este resultado, o Rio Ave é 9ª posição da Liga Portuguesa com 27 pontos e está no 9º lugar com 22 pontos. \*

	1	2	3	4	5	
Falta informação chave, o que torna a frase muito incompleta.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Toda a informação chave é apresentada, sendo assim a frase é adequada.

BACK

NEXT

 Page 4 of 5

Figure A.9: Part 9 of questionnaire one.

**Belenenses 0 - 0 Braga**

<https://www.zerozero.pt/jogo.php?id=5003898>

**Expulsão (Fábio Nunes, 84') - Inteligibilidade**

Destaque para Fábio Nunes que foi expulso por acumulação de amarelos, deixando o Belenenses jogar com 10. \*

	1	2	3	4	5	
A frase é imperceptível.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	A frase é lexicalmente e gramaticalmente correta.

Até final, nota para a expulsão de Fábio Nunes, do Belenenses, aos 84' minutos, deixando o Belenenses a partir do Belenenses, aos 84' minutos. \*

	1	2	3	4	5	
A frase é imperceptível.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	A frase é lexicalmente e gramaticalmente correta.

Nota ainda para a expulsão de Fábio Nunes. \*

	1	2	3	4	5	
A frase é imperceptível.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	A frase é lexicalmente e gramaticalmente correta.

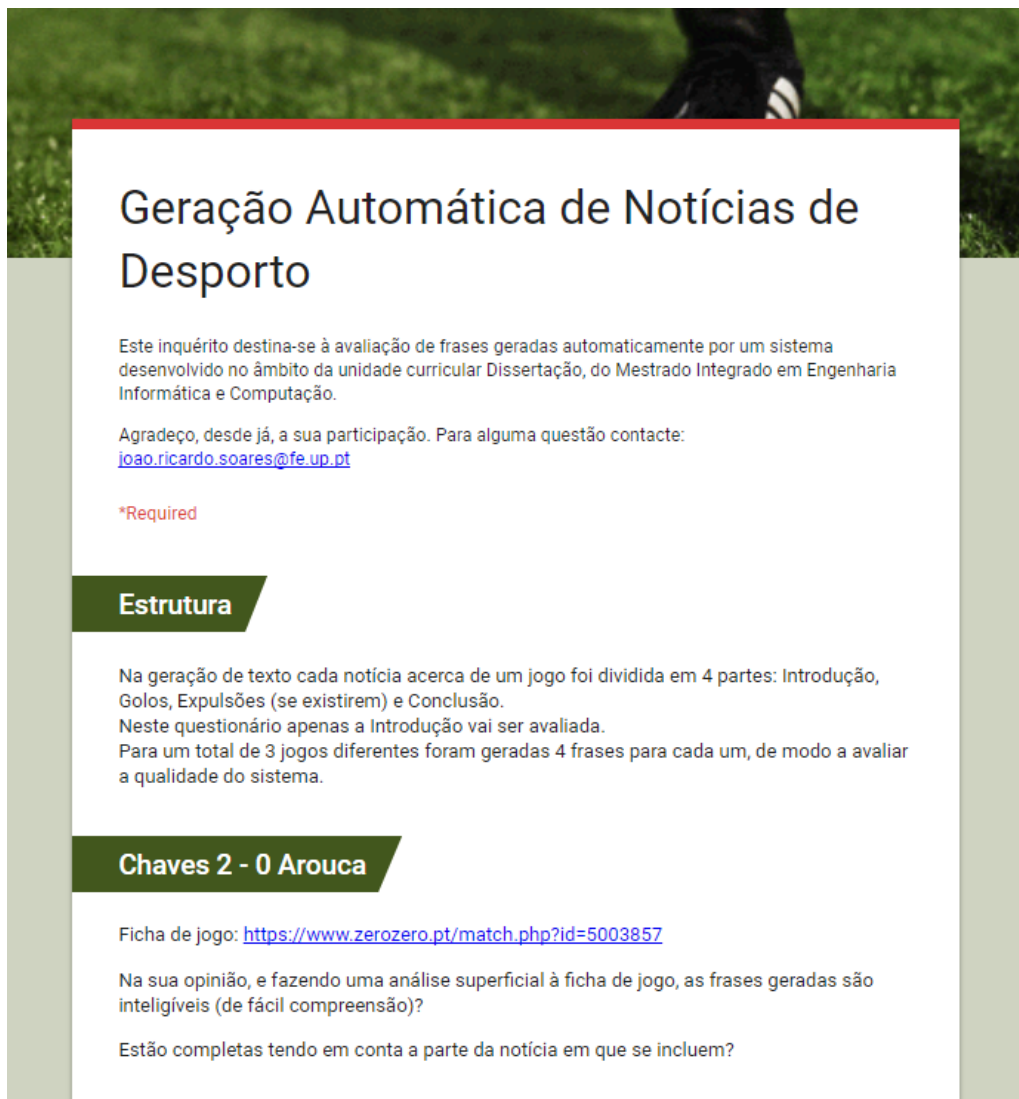
Figure A.10: Part 10 of questionnaire one.



## Appendix B

### Questionnaire two

In this appendix are presented images from the questionnaire two.



**Geração Automática de Notícias de Desporto**

Este inquérito destina-se à avaliação de frases geradas automaticamente por um sistema desenvolvido no âmbito da unidade curricular Dissertação, do Mestrado Integrado em Engenharia Informática e Computação.

Agradeço, desde já, a sua participação. Para alguma questão contacte:  
[joao.ricardo.soares@fe.up.pt](mailto:joao.ricardo.soares@fe.up.pt)

\*Required

**Estrutura**

Na geração de texto cada notícia acerca de um jogo foi dividida em 4 partes: Introdução, Golos, Expulsões (se existirem) e Conclusão.  
Neste questionário apenas a Introdução vai ser avaliada.  
Para um total de 3 jogos diferentes foram geradas 4 frases para cada um, de modo a avaliar a qualidade do sistema.

**Chaves 2 - 0 Arouca**

Ficha de jogo: <https://www.zerozero.pt/match.php?id=5003857>

Na sua opinião, e fazendo uma análise superficial à ficha de jogo, as frases geradas são inteligíveis (de fácil compreensão)?

Estão completas tendo em conta a parte da notícia em que se incluem?

Figure B.1: Part 1 of questionnaire two.

### Introdução - Inteligibilidade

O Chaves recebeu e venceu o Arouca por 2 x 0, em jogo da 22ª jornada da Liga Portuguesa. \*

	1	2	3	4	5	
A frase é imperceptível.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	A frase é lexicalmente e gramaticalmente correta.

O Arouca foi derrotado pelo Chaves por 2 x 0, este Sábado. \*

	1	2	3	4	5	
A frase é imperceptível.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	A frase é lexicalmente e gramaticalmente correta.

O Arouca perdeu por Ricardo Soares, por 2 x 0, este sábado, o Arouca, este sábado, em encontro da 22ª jornada da Liga Portuguesa. \*

	1	2	3	4	5	
A frase é imperceptível.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	A frase é lexicalmente e gramaticalmente correta.

O Chaves bateu o Arouca por 2 x 0. \*

	1	2	3	4	5	
A frase é imperceptível.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	A frase é lexicalmente e gramaticalmente correta.

Figure B.2: Part 2 of questionnaire two.

### Introdução - Completitude

O Chaves recebeu e venceu o Arouca por 2 x 0, em jogo da 22ª jornada da Liga Portuguesa. \*

	1	2	3	4	5	
Falta informação chave, o que torna a frase muito incompleta.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Toda a informação chave é apresentada, sendo assim a frase é adequada.

O Arouca foi derrotado pelo Chaves por 2 x 0, este Sábado. \*

	1	2	3	4	5	
Falta informação chave, o que torna a frase muito incompleta.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Toda a informação chave é apresentada, sendo assim a frase é adequada.

O Arouca perdeu por Ricardo Soares, por 2 x 0, este sábado, o Arouca, este sábado, em encontro da 22ª jornada da Liga Portuguesa. \*

	1	2	3	4	5	
Falta informação chave, o que torna a frase muito incompleta.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Toda a informação chave é apresentada, sendo assim a frase é adequada.

Figure B.3: Part 3 of questionnaire two.

## Questionnaire two

O Chaves bateu o Arouca por 2 x 0. \*

	1	2	3	4	5	
Falta informação chave, o que torna a frase muito incompleta.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Toda a informação chave é apresentada, sendo assim a frase é adequada.

NEXT

Page 1 of 3

Figure B.4: Part 4 of questionnaire two.

## Geração Automática de Notícias de Desporto

\*Required

### Tondela 0 - 1 Feirense

Ficha de jogo: <http://www.zerozero.pt/match.php?id=5003851>

#### Introdução - Inteligibilidade

A equipa de Nuno Manta venceu o Tondela, em jogo referente à 21ª jornada. \*

	1	2	3	4	5	
A frase é imperceptível.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	A frase é lexicalmente e gramaticalmente correta.

O Feirense venceu o Tondela em jogo a contar para a 21ª jornada da Liga Portuguesa, este Sábado, em encontro da 21ª jornada da Liga Portuguesa. \*

	1	2	3	4	5	
A frase é imperceptível.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	A frase é lexicalmente e gramaticalmente correta.

O Feirense foi ao terreno do Tondela vencer por 0 x 1. \*

	1	2	3	4	5	
A frase é imperceptível.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	A frase é lexicalmente e gramaticalmente correta.

Figure B.5: Part 5 of questionnaire two.

## Questionnaire two

O Feirense foi ao terreno do Tondela vencer por 0 x 1. \*

	1	2	3	4	5	
A frase é imperceptível.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	A frase é lexicalmente e gramaticalmente correta.

O Feirense foi ao terreno do Tondela vencer por 1 bola a 0, este Sábado, em encontro da 21.<sup>a</sup> jornada da Liga Portuguesa. \*

	1	2	3	4	5	
A frase é imperceptível.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	A frase é lexicalmente e gramaticalmente correta.

**Introdução - Completitude**

A equipa de Nuno Manta venceu o Tondela, em jogo referente à 21.<sup>a</sup> jornada. \*

	1	2	3	4	5	
Falta informação chave, o que torna a frase muito incompleta.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Toda a informação chave é apresentada, sendo assim a frase é adequada.

Figure B.6: Part 6 of questionnaire two.

## Questionnaire two

O Feirense venceu o Tondela em jogo a contar para a 21.<sup>a</sup> jornada da Liga Portuguesa, este Sábado, em encontro da 21.<sup>a</sup> jornada da Liga Portuguesa. \*

	1	2	3	4	5	
Falta informação chave, o que torna a frase muito incompleta.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Toda a informação chave é apresentada, sendo assim a frase é adequada.

O Feirense foi ao terreno do Tondela vencer por 0 x 1. \*

	1	2	3	4	5	
Falta informação chave, o que torna a frase muito incompleta.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Toda a informação chave é apresentada, sendo assim a frase é adequada.

O Feirense foi ao terreno do Tondela vencer por 1 bola a 0, este Sábado, em encontro da 21.<sup>a</sup> jornada da Liga Portuguesa. \*

	1	2	3	4	5	
Falta informação chave, o que torna a frase muito incompleta.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Toda a informação chave é apresentada, sendo assim a frase é adequada.

BACK

NEXT

Page 2 of 3

Figure B.7: Part 7 of questionnaire two.

## Geração Automática de Notícias de Desporto

\*Required

### Boavista 1 - 1 Braga

Ficha de jogo: <https://www.zerozero.pt/match.php?id=5003853>

#### Introdução - Inteligibilidade

Boavista e Braga dividiram os pontos após um empate a 1 golo.

\*

	1	2	3	4	5	
A frase é imperceptível.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	A frase é lexicalmente e gramaticalmente correta.

Este Domingo, na deslocação ao Boavista, a contar para a 21ª jornada da 21ª jornada da Liga Portuguesa. \*

	1	2	3	4	5	
A frase é imperceptível.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	A frase é lexicalmente e gramaticalmente correta.

O Boavista, de Miguel Leal, não foi além de um empate a 1, em partida referente à 21ª jornada. \*

	1	2	3	4	5	
A frase é imperceptível.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	A frase é lexicalmente e gramaticalmente correta.

Figure B.8: Part 8 of questionnaire two.



## Questionnaire two

Braga empataram a 1, este Domingo. \*

	1	2	3	4	5	
A frase é imperceptível.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	A frase é lexicalmente e gramaticalmente correta.

### Introdução - Completitude

Boavista e Braga dividiram os pontos após um empate a 1 golo. \*

	1	2	3	4	5	
Falta informação chave, o que torna a frase muito incompleta.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Toda a informação chave é apresentada, sendo assim a frase é adequada.

Este Domingo, na deslocação ao Boavista, a contar para a 21ª jornada da 21ª jornada da Liga Portuguesa. \*

	1	2	3	4	5	
Falta informação chave, o que torna a frase muito incompleta.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Toda a informação chave é apresentada, sendo assim a frase é adequada.

Figure B.9: Part 9 of questionnaire two.

## Questionnaire two

O Boavista, de Miguel Leal, não foi além de um empate a 1, em partida referente à 21ª jornada. \*

	1	2	3	4	5	
Falta informação chave, o que torna a frase muito incompleta.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Toda a informação chave é apresentada, sendo assim a frase é adequada.

Braga empataram a 1, este Domingo. \*

	1	2	3	4	5	
Falta informação chave, o que torna a frase muito incompleta.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Toda a informação chave é apresentada, sendo assim a frase é adequada.

[BACK](#) [SUBMIT](#) Page 3 of 3

Figure B.10: Part 10 of questionnaire two.

# References

- [ABD00] Hiyan Alshawhi, Srinivas Bangalore, and Shona Douglas. Learning dependency translation models as collections of finite-state head transducers. *Computational Linguistics*, 26(1):45–60, 2000.
- [Air16] João Pinto Barbosa Machado Aires. Automatic generation of sports news. 2016.
- [BDM<sup>+</sup>92] Peter F Brown, Peter V Desouza, Robert L Mercer, Vincent J Della Pietra, and Jenifer C Lai. Class-based n-gram models of natural language. *Computational linguistics*, 18(4):467–479, 1992.
- [Bec09] Sean Bechhofer. Owl: Web ontology language. In *Encyclopedia of Database Systems*, pages 2008–2009. Springer, 2009.
- [Bel09] Anja Belz. That’s nice... what can you do with it? *Computational Linguistics*, 35(1):111–118, 2009.
- [BK03] Jeff A Bilmes and Katrin Kirchhoff. Factored language models and generalized parallel backoff. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology: companion volume of the Proceedings of HLT-NAACL 2003—short papers-Volume 2*, pages 4–6. Association for Computational Linguistics, 2003.
- [BKL09] Steven Bird, Ewan Klein, and Edward Loper. *Natural language processing with Python: analyzing text with the natural language toolkit*. " O’Reilly Media, Inc.", 2009.
- [Bod93] Rens Bod. Using an annotated corpus as a stochastic grammar. In *Proceedings of the sixth conference on European chapter of the Association for Computational Linguistics*, pages 37–44. Association for Computational Linguistics, 1993.
- [BR06] Anja Belz and Ehud Reiter. Comparing automatic and human evaluation of nlg systems. In *EACL*, 2006.
- [CG96] Stanley F Chen and Joshua Goodman. An empirical study of smoothing techniques for language modeling. In *Proceedings of the 34th annual meeting on Association for Computational Linguistics*, pages 310–318. Association for Computational Linguistics, 1996.
- [Coc96] José Coch. Evaluating and comparing three text-production techniques. In *Proceedings of the 16th conference on Computational linguistics-Volume 1*, pages 249–254. Association for Computational Linguistics, 1996.
- [Dav74] Anthony Davey. The formalisation of discourse production. 1974.

## REFERENCES

- [DIB<sup>+</sup>11] Myroslava O Dzikovska, Amy Isard, Peter Bell, Johanna D Moore, Natalie Steinhäuser, and Gwendolyn Campbell. Beetle ii: an adaptable tutorial dialogue system. In *Proceedings of the SIGDIAL 2011 Conference*, pages 338–340. Association for Computational Linguistics, 2011.
- [dNP13] Eder Miranda de Novais and Ivandré Paraboni. Portuguese text generation using factored language models. *Journal of the Brazilian Computer Society*, 19(2):135–146, 2013.
- [dNPF11] Eder de Novais, Ivandré Paraboni, and Diogo Ferreira. Highly-inflected language generation using factored language models. *Computational Linguistics and Intelligent Text Processing*, pages 429–438, 2011.
- [Fie05] Armin Fiedler. Natural language proof explanation. In *Mechanizing Mathematical Reasoning*, pages 342–363. Springer, 2005.
- [FPRL06] Leo Ferres, Avi Parush, Shelley Roberts, and Gitte Lindgaard. Helping people with visual impairments gain access to graphical information through natural language: The igrph system. In *International Conference on Computers for Handicapped Persons*, pages 1122–1130. Springer, 2006.
- [GA07] Dimitrios Galanis and Ion Androutsopoulos. Generating multilingual descriptions from linguistically annotated owl ontologies: the naturalowl system. In *Proceedings of the Eleventh European Workshop on Natural Language Generation*, pages 143–146. Association for Computational Linguistics, 2007.
- [GDK94] Eli Goldberg, Norbert Driedger, and Richard I Kittredge. Using natural-language processing to produce weather forecasts. *IEEE Expert*, 9(2):45–53, 1994.
- [GJ93] Julia Rose Galliers and K Sparck Jones. Evaluating natural language processing systems. 1993.
- [Gom10] Maarten Van Gompel. Pynlpl - python natural language processing library. <https://github.com/proycon/pynlpl>, 2010.
- [GR03] Yoshihiko Gotoh and Steve Renals. Statistical language modelling. In *Text-and Speech-Triggered Information Access*, pages 78–105. Springer, 2003.
- [GR09] Albert Gatt and Ehud Reiter. Simplenlg: A realisation engine for practical applications. In *Proceedings of the 12th European Workshop on Natural Language Generation*, pages 90–93. Association for Computational Linguistics, 2009.
- [HBS<sup>+</sup>12] Abram Hindle, Earl T. Barr, Zhendong Su, Mark Gabel, and Premkumar Devanbu. On the naturalness of software. In *Proceedings of the 34th International Conference on Software Engineering, ICSE ’12*, pages 837–847, Piscataway, NJ, USA, 2012. IEEE Press.
- [HM85] MAK Halliday and CM Matthiessen. An introduction to functional grammar. edward arnold, london. *Australian Rev. Appl. Linguist*, 10(2):163–181, 1985.
- [JJ00] Daniel Jurafsky and H James. Speech and language processing an introduction to natural language processing, computational linguistics, and speech. 2000.

## REFERENCES

- [Jon01] K Sparck Jones. Natural language processing: a historical review. *University of Cambridge*, pages 2–10, 2001.
- [JS97] Aravind K Joshi and Yves Schabes. Tree-adjoining grammars. In *Handbook of formal languages*, pages 69–123. Springer, 1997.
- [KBC<sup>+</sup>09] Alexander Koller, Donna Byron, Justine Cassell, Robert Dale, Johanna Moore, Jon Oberlander, and Kristina Striegnitz. The software architecture for the first challenge on generating instructions in virtual environments. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics: Demonstrations Session*, pages 33–36. Association for Computational Linguistics, 2009.
- [KPG86] Richard Kittredge, Alain Polguere, and Eli Goldberg. Synthesizing weather forecasts from formatted data. In *Proceedings of the 11th conference on Computational linguistics*, pages 563–565. Association for Computational Linguistics, 1986.
- [Kuk83] Karen Kukich. Design of a knowledge-based report generator. In *Proceedings of the 21st annual meeting on Association for Computational Linguistics*, pages 145–150. Association for Computational Linguistics, 1983.
- [Lan00] Irene Langkilde. Forest-based statistical sentence generation. In *Proceedings of the 1st North American chapter of the Association for Computational Linguistics conference*, pages 170–177. Association for Computational Linguistics, 2000.
- [LB02] Edward Loper and Steven Bird. Nltk: The natural language toolkit. In *Proceedings of the ACL-02 Workshop on Effective tools and methodologies for teaching natural language processing and computational linguistics-Volume 1*, pages 63–70. Association for Computational Linguistics, 2002.
- [LCFQY14] Nathalie Rose Lim-Cheng, Gabriel Isidro G Fabia, MEG Quebral, and MT Yu. Shed: An online diet counselling system. In *DLSU Research Congress*, 2014.
- [Lev12] Steven Levy. Can an algorithm write a better news story than a human reporter? *Wired*, 24:2012, 2012. <https://www.wired.com/2012/04/can-an-algorithm-write-a-better-news-story-than-a-human-reporter/> (visited: 2016-11-12).
- [LK98] Irene Langkilde and Kevin Knight. Generation that exploits corpus-based statistical knowledge. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics-Volume 1*, pages 704–710. Association for Computational Linguistics, 1998.
- [LS99] Bret Larget and Donald L Simon. Markov chain monte carlo algorithms for the bayesian analysis of phylogenetic trees. *Molecular biology and evolution*, 16(6):750–759, 1999.
- [MJHL16] Elena Manishina, Bassam Jabaian, Stéphane Huet, and Fabrice Lefevre. Automatic corpus extension for data-driven natural language generation. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016), Paris, France. European Language Resources Association (ELRA)*, 2016.

## REFERENCES

- [MT88] William C Mann and Sandra A Thompson. Rhetorical structure theory: Toward a functional theory of text organization. *Text-Interdisciplinary Journal for the Study of Discourse*, 8(3):243–281, 1988.
- [MW08] Simon Mille and Leo Wanner. Multilingual summarization in practice: the case of patent claims. In *Proceedings of the 12th European association of machine translation conference*, pages 120–129, 2008.
- [PRG<sup>+</sup>09] François Portet, Ehud Reiter, Albert Gatt, Jim Hunter, Somayajulu Sripada, Yvonne Freer, and Cindy Sykes. Automatic generation of textual summaries from neonatal intensive care data. *Artificial Intelligence*, 173(7-8):789–816, 2009.
- [PVG<sup>+</sup>11] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. Scikit-learn: Machine learning in python. *Journal of Machine Learning Research*, 12(Oct):2825–2830, 2011.
- [RD97] Ehud Reiter and Robert Dale. Building applied natural language generation systems. *Natural Language Engineering*, 3(01):57–87, 1997.
- [RM96] Jacques Robin and Kathleen McKeown. Empirically designing and evaluating a new revision-based model for summary generation. *Artificial Intelligence*, 85(1):135–179, 1996.
- [Roa01] Brian Roark. Probabilistic top-down parsing and language modeling. *Computational linguistics*, 27(2):249–276, 2001.
- [Roo14] Kevin Roose. Robots are invading the news business, and it’s great for journalists. *New York*, 11, 2014. <http://nymag.com/daily/intelligencer/2014/07/why-robot-journalism-is-great-for-journalists.html> (visited: 2016-11-12).
- [RRO03] Ehud Reiter, Roma Robertson, and Liesl M Osman. Lessons from a failure: Generating tailored smoking cessation letters. *Artificial Intelligence*, 144(1-2):41–58, 2003.
- [RSBB16] Alejandro Ramos-Soto, Alberto Bugarín, and Senén Barro. On the role of linguistic descriptions of data in the building of natural language generation systems. *Fuzzy Sets and Systems*, 285:31–51, 2016.
- [RSH<sup>+</sup>05] Ehud Reiter, Somayajulu Sripada, Jim Hunter, Jin Yu, and Ian Davy. Choosing words in computer-generated weather forecasts. *Artificial Intelligence*, 167(1-2):137–169, 2005.
- [RTA<sup>+</sup>09] Ehud Reiter, Ross Turner, Norman Alm, Rolf Black, Martin Dempster, and Annalu Waller. Using nlg to help language-impaired users tell stories and participate in social dialogues. In *Proceedings of the 12th European Workshop on Natural Language Generation*, pages 1–8. Association for Computational Linguistics, 2009.
- [S<sup>+</sup>02] Andreas Stolcke et al. Srilm-an extensible language modeling toolkit. In *Interspeech*, volume 2002, page 2002, 2002.

## REFERENCES

- [SB09] Christina Sauper and Regina Barzilay. Automatically generating wikipedia articles: A structure-aware approach. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 1-Volume 1*, pages 208–216. Association for Computational Linguistics, 2009.
- [SB11] Mark Steedman and Jason Baldridge. Combinatory categorial grammar. *Non-Transformational Syntax: Formal and Explicit Models of Grammar*. Wiley-Blackwell, 2011.
- [Swa77] William R Swartout. A digitalis therapy advisor with explanations. In *Proceedings of the 5th international joint conference on Artificial intelligence-Volume 2*, pages 819–825. Morgan Kaufmann Publishers Inc., 1977.
- [TKdP<sup>+</sup>01] Mariët Theune, Esther Klabbers, Jan-Roelof de Pijper, Emiel Krahmer, and Jan Odijk. From data to speech: a general approach. *Natural Language Engineering*, 7(01):47–86, 2001.
- [VBP<sup>+</sup>15] Marta Vicente, Cristina Barros, Fernando S Peregrino, Francisco Agulló, and Elena Lloret. La generacion de lenguaje natural: análisis del estado actual. *Computación y Sistemas*, 19(4):721–756, 2015.
- [WCM10] Michael White, Robert AJ Clark, and Johanna D Moore. Generating tailored, comparative descriptions with contextually appropriate intonation. *Computational Linguistics*, 36(2):159–201, 2010.
- [WGK<sup>+</sup>15] Tsung-Hsien Wen, Milica Gasic, Dongho Kim, Nikola Mrksic, Pei-Hao Su, David Vandyke, and Steve Young. Stochastic language generation in dialogue using recurrent neural networks with convolutional sentence reranking. *arXiv preprint arXiv:1508.01755*, 2015.
- [Whi12] Michael White. Openccg realizer manual. *Documentation of the OpenCCG Realizer*, 2012.
- [WR08] Sandra Williams and Ehud Reiter. Generating basic skills reports for low-skilled readers. *Natural Language Engineering*, 14(04):495–525, 2008.
- [YRHM07] Jin Yu, Ehud Reiter, Jim Hunter, and Chris Mellish. Choosing the content of textual summaries of large time-series data sets. *Natural Language Engineering*, 13(01):25–49, 2007.
- [ZM65] AK Zholkovskii and IA Mel’chuk. On a possible method and instrument for semantic synthesis. *Nauchno-tekhnicheskaya informatsiya*, (6), 1965.