YAN VIANNA SYM

AN AUTOMATED JOURNALISM AGENT COVERING THE BLUE AMAZON

São Paulo
2023

YAN VIANNA SYM

AN AUTOMATED JOURNALISM AGENT COVERING THE BLUE AMAZON

Dissertation presented to Escola Politécnica of Universidade de São Paulo to obtain Master of Sciences degree in Eletrical Engineering.

São Paulo
2023

YAN VIANNA SYM

AN AUTOMATED JOURNALISM AGENT COVERING THE BLUE AMAZON

Corrected Version

Dissertation presented to Escola Politécnica of Universidade de São Paulo to obtain Master of Sciences degree in Eletrical Engineering.

Concentration Area

Computer Engineering

Advisor:

Fabio Gagliardi Cozman

São Paulo
2023

Este exemplar foi revisado e corrigido em relação à versão original, sob responsabilidade única do autor e com a anuência de seu orientador.

São Paulo, 10 de agosto de 2023

Assinatura do autor: 

Assinatura do orientador: 

Catalogação-na-publicação

Sym, Yan
Um Agente de Jornalismo Automatizado Cobrindo a Amazônia Azul / Y. Sym -- versão corr. -- São Paulo, 2023.
70 p.

Dissertação (Mestrado) - Escola Politécnica da Universidade de São Paulo. Departamento de Engenharia de Computação e Sistemas Digitais.

1.Geração de Linguagem Natural 2.Jornalismo Automatizado 3.Transformers 4.Amazônia Azul I.Universidade de São Paulo. Escola Politécnica. Departamento de Engenharia de Computação e Sistemas Digitais II.t.

"Strategy without tactics is the slowest route to victory. Tactics without strategy is the noise before defeat" — Sun-Tzu, The Art of War

ACKNOWLEDGMENTS

We would like to thank the Center for Artificial Intelligence (C4AI: www.c4ai.inova.usp) with support from the São Paulo Research Foundation (FAPESP grant #2019/07665-4) and from IBM Corporation.

RESUMO

A Amazônia Azul é a Zona Econômica Exclusiva do Brasil (ZEE), com uma área total de 3,6 milhões de quilômetros quadrados ao longo da costa brasileira, rica em biodiversidade marinha e recursos energéticos. Poucas informações sobre esta área são compartilhadas com o público em geral, principalmente porque os dados disponíveis estão armazenados em múltiplas localizações e em formatos distintos. Para resolver esse problema, apresentamos nosso repórter da Amazônia Azul, um agente autônomo que utiliza técnicas de robô-jornalismo e geração de linguagem natural para publicar diariamente relatórios e curiosidades sobre a Amazônia Azul em português. Ao coletar, armazenar e analisar dados estruturados de diversas fontes, o agente aplica conhecimento do domínio para gerar, validar e publicar conteúdo no Twitter. Também comparamos as arquiteturas mais utilizadas para gerar textos a partir de dados estruturados, analisando os resultados obtidos e identificando as principais vantagens e desvantagens de cada arquitetura. Código e corpus estão disponíveis publicamente.

Palavras-Chave – Geração de Linguagem Natural, Robô Jornalismo, Transformers, Amazônia Azul

ABSTRACT

The Blue Amazon is Brazil's exclusive economic zone (EEZ), with an offshore area of 3.6 million square kilometers along the Brazilian coast, rich in marine biodiversity and energy resources. Little information about this area is shared with the general public, mainly because the available data is stored in multiple locations and different formats. To address this problem, we present our Blue Amazon reporter, an autonomous agent which applies automated journalism and natural language generation algorithms to publish daily reports and trivia about the Blue Amazon in Brazilian Portuguese. By collecting, storing, and analyzing structured data from multiple sources, the agent applies domain knowledge to generate, validate, and publish content on Twitter. We also compare the most frequently employed architectures to generate texts from structured data, analyzing the obtained results and identifying the main advantages and disadvantages of each architecture. Code and corpus are publicly available.

Keywords – Natural Language Generation, Automated Journalism, Transformers, Blue Amazon

LIST OF FIGURES

LIST OF TABLES

LIST OF ACRONYMS

API - Application Programming Interface

BERT - Bidirectional Encoder Representations from Transformers

BLAB - BLue Amazon Brain

BLEU - Bilingual Evaluation Understudy

BLEURT - Bilingual Evaluation Understudy with Representations from Transformers

C4AI - Center for Artificial Intelligence

COMET - Crosslingual Optimized Metric for Evaluation of Translation

EEZ - Exclusive Economic Zone

GAN - Generative Adversarial Network

GLEU - Automatic Evaluation of Sentence-Level Fluency

GPT - Generative Pre-trained Transformer

GPU - Graphics Processing Unit

LSTM - Long short-term memory

METEOR - Metric for Evaluation of Translation with Explicit ORdering

NLG - Natural Language Generation

NLP - Natural Language Processing

NLU - Natural Language Understanding

PRISM - Probability is the metric

ROUGE - Recall-Oriented Understudy for Gisting Evaluation

T5 - Text-to-Text Transfer Transformer

USP - University of São Paulo

CONTENTS

1 INTRODUCTION

Natural Language Generation (NLG) is a field at the intersection of linguistics, computer science, and artificial intelligence, concerned with generating readable, coherent and meaningful explanatory text or speech so as to describe non-linguistic input data (REITER; DALE, 2000). NLG is often viewed as complementary to Natural Language Understanding (NLU) and part of Natural Language Processing (NLP). Whereas in NLU the goal is to understand input sentences to produce machine representations, in NLG the system must make decisions about how to transform representations into meaningful words and phrases (LIDDY, 2001).

Most efforts in NLG are currently guided by a sentence generation paradigm, which follows a data-to-text approach. Successful examples of data-to-text systems can be found in weather forecasting (SRIPADA et al., 2004), financial and analytical reporting (NESTERENKO, 2016), summarization of statistical data (HARTLEY; PARIS, 1996), industrial monitoring (KIM; BAE; AN, 2020), conversational agents (PARMAR et al., 2019). Amongst NLG applications, robot journalism is one of the most prominent endeavors thanks to the abundance of structured data streams available today, thus allowing automated systems to report recurring material with high-fidelity and lexical variation (GRAEFE, 2016). Robot journalism refers to the generation of stories by algorithms based on input data and the process to automatically publish text without human intervention. Although the process of generating news content is often referred to as robot journalism, other terms such as computational journalism and automated journalism are also often utilized (FIRAT, 2019).

Traditionally, most automated journalism systems have been designed in a modular fashion, as this facilitates reuse in different domains (GATT; KRAHMER, 2018). In such systems, non-linguistic input data is converted into natural language through several explicit intermediate transformations and sequential tasks related to content selection, sentence planning and linguistic realization (FERREIRA et al., 2019). The two most frequently used automated journalism architectures are the template-based approach,

which is application-dependent and lacks generalization capabilities due to its rule-based nature, and the pipeline-based approach, which embodies linguistic insights to convert data to text by applying a series of sequential steps. These steps can be performed in several manners, for example applying domain specific heuristics or using deep learning models.

Usually, template-based systems resort to a small amount of textual templates in order to generate texts. This process in performed in two phases: Content Selection, where the information to be verbalized is selected, and Text Realization, where the selected values are inserted into pre-customized textual templates. For example, a simple template-based system might start out with information about a new earthquake with a magnitude of 1.7 mR and depth of 10km, detected by the Seismology Center of the University of São Paulo, in the city of Arapiraca, Alagoas (AL):

```
EARTHQUAKE(city="Arapiraca", uf="AL", magnitude="1.7mR", depth="10km", entity="Seismology
Center of the University of São Paulo")
```

This intent-attribute-value input is then directly associated with an output text, and the gaps are filled by looking up the relevant information in a table. An example of template for this scenario is:

```
A new earthquake was detected in [location] with a magnitude of [magnitude] and depth of
[depth], by the [entity].  Stay safe!
```

In contrast, the NLG pipeline approach converts structured input data to output text by relying on document planning and following a series of sequential steps. Figure 1 shows a pipeline architecture with an example in the case of our automated journalism agent. The system receives, as input, information regarding location, weather and vessels, and outputs the following text (here translated to English):

Good Morning! Today in Rio de Janeiro (RJ) the weather is sunny, and the average expected temperature during the day is 32°C. Currently, 280 fishing vessels are in the port, and this is the highest number of vessels reported in the last 6 months. According to the Marine Traffic website, this phenomenon may have been caused by the excellent fishing conditions.

The first step of the pipeline approach is to identify the relevant information, analyzing non-linguistic data to determine the specific attributes or variables that are relevant

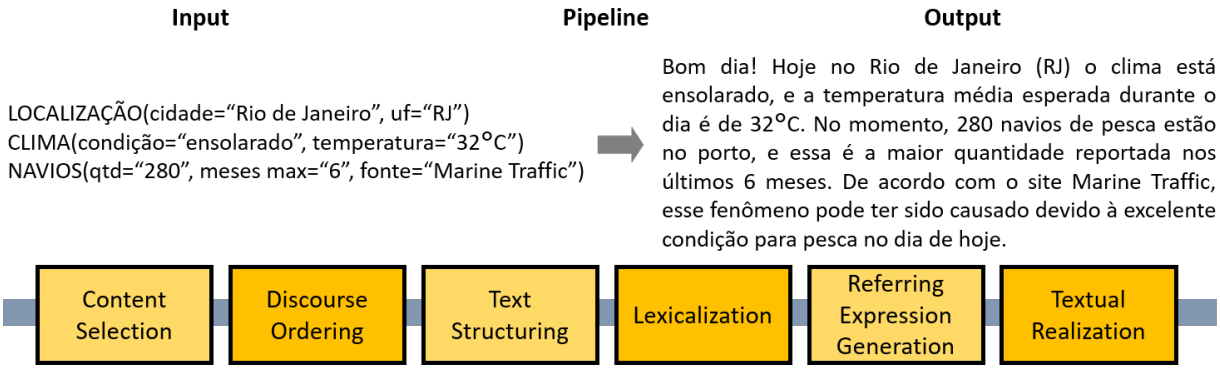

Figure 1: Pipeline architecture with an example of a generated text for our automated journalism agent.

for the domain. For example, if the input data is about movies, relevant attributes might include title, genre, director, and actors. The next step is to determine the logical flow or structure of the generated text, which can be done by identifying the relationships and dependencies between different pieces of information, are organizing them into sentences and paragraphs. For example, in a movie description, it would be logical to mention the title before discussing the genre or director. Finally, both syntactic and morphological adjustments are performed to transform intermediate machine representations into grammatically consistent text.

The emergence of neural-based NLG systems in recent years has changed the field: provided there is enough labeled data for training a machine learning model, learning a direct mapping from structured input to textual output has become reality (LI, 2017). This has led to the recent development of deep learning end-to-end models, which directly learn input-output mappings and rely far less on explicit intermediary representations and linguistic insights.

Interesting domains for automated journalism are ocean monitoring, climate change and environmental sustainability. The ocean is severely damaged, and if current trends continue, there will be disastrous consequences for the earth as it is essential to halting climate change, fostering economic growth, and preserving biodiversity (COSTA; GONÇALVES; GONÇALVES, 2022). In March 2021, the Suez Canal made headlines around the world when a 400-meter-long container ship called the Ever Given ran aground and blocked the canal for nearly a week, causing a major traffic jam with hundreds of ships on either side. The incident highlighted the importance of the canal for global trade and raised questions about its vulnerability to disruption. The Suez Canal is an artificial sea-level waterway located in Egypt that connects the Mediterranean Sea to the Red Sea.

Accurate and low latency information reports can be very helpful in these situations, but communicating to general audiences in a accessible way usually demands coverage by specialized human journalists.

To address this issue, we present a robot journalism agent, called BLAB Reporter, which was built using a mix of NLG architectures and automated journalism concepts [1]. Our system generates daily reports and news about the Blue Amazon and publishes them on Twitter in Brazilian Portuguese.

The Blue Amazon is a vast region of the Atlantic Ocean as shown in Figure 2. Also referred to as the Exclusive Economic Zone (EEZ) of Brazil, it is a region rich in marine biodiversity and energy resources with an offshore area of 3.6 million square kilometers, and is the third largest EEZ in the world, after the EEZs of the United States and France (WIESEBRON, 2013). Within Brazil's EEZ, the country has exclusive rights to explore and exploit the natural resources found in the ocean, including fish stocks, oil and gas reserves, and minerals. Brazil also has the responsibility of conserving and managing the marine environment in this area. The importance of the Blue Amazon lies in its enormous potential for sustainable development, having unique beauty and contributing to tourism and other industries. For instance, the Abrolhos archipelago, that concentrates the largest marine biodiversity of the South Atlantic, contains about 1,300 registered species (LEÃO; KIKUCHI, 2001).

The Brazilian government has implemented several policies and programs to protect the marine environment within its EEZ, including efforts to fight illegal fishing, reduce marine pollution, and promote sustainable use of marine resources. The expression "Blue Amazon" was coined by the Brazilian navy to emphasize the importance of the territory in a comparison with the largest tropical forest in the world (CASTRO et al., 2017). The "BLue Amazon Brain" (BLAB) is a project that integrates a number of services aimed at disseminating information about this region and its importance BLAB includes arguments, causes, explanations, reasoning, and plans for specific tasks, providing answers to the most diverse questions about the marine ecosystem (PIROZELLI et al., 2022). The project is hosted by C4AI (Center for Artificial Intelligence), and it is supported by FAPESP and IBM for research in Artificial Intelligence.

Even though it is technically feasible to use neural end-to-end methods in real world applications, this does not necessarily mean that they are superior to other approaches in every scenario. Empirical studies have demonstrated that a combination of template

[1] ⟨https://github.com/C4AI/blab-reporter⟩

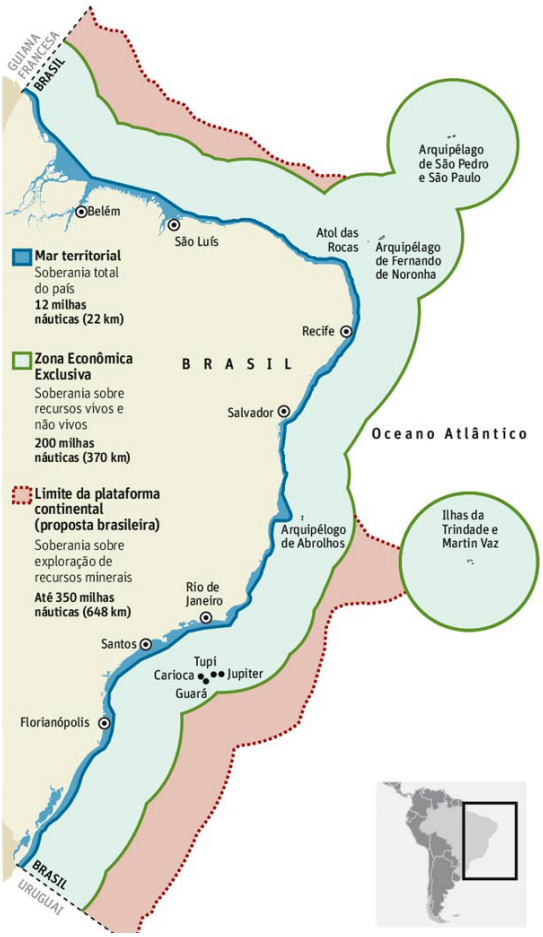

Figure 2: The Blue Amazon (extracted from Wikipedia, in public domain).

and pipeline systems often produce texts that are more appropriate than neural-based approaches, which frequently hallucinate data or content unsupported by the semantic input (FERREIRA et al., 2019). For the particular task of automated journalism, reporting inaccurate data would seriously undermine a robot's credibility and could have serious implications on sensitive domains, such as environmental and clinical reports. Modular-based approaches also have the advantage of allowing for auditing, while neural end-to-end approaches behave as black-boxes (CAMPOS et al., 2020). Our automated journalism agent combines multiple NLG approaches in order to generate fluent and informative texts about the Blue Amazon and publish them on Twitter [2]. Figure 3 shows an example of tweet about the weather in in Buzios (RJ), published by our Blue Amazon agent in March 2023.

[2] ⟨https://twitter.com/BLAB_Reporter⟩

Figure 3: Example of a tweet generated by our Blue Amazon agent.

1.1 Objectives

The main contributions of this work are:

- The construction of the first publicly available Brazilian Portuguese NLG dataset containing information about the Blue Amazon. A corpus of verbalizations of non-linguistic data was created based on syntactical and lexical patterning abstracted from data collected from publicly available sources. Intermediate representations were annotated for each entry in order to develop our corpus [3]. A combination of automatic and human evaluation together with a qualitative analysis was then carried out to measure the fluency, semantics and lexical variety of the generated texts.

- A NLG application, the BLAB Reporter, that collects and stores data from multiple publicly available sources and that combines different automated journalism architectures to publish daily reports, news and trivia about the Blue Amazon on Twitter.

- A comparison between the three most commonly used automated journalism architectures: template, pipeline and neural end-to-end. We collected and stored

[3] ⟨https://github.com/C4AI/blab-reporter/tree/main/experiments/corpus⟩

information from the Blue Amazon during multiple days and applied different NLG approaches to generate text from data. We utilized a combination of automatic and human metrics to evaluate the pros and cons of each architecture, and discussed when they should or shouldn't be used.

1.2 Organization of the Manuscript

This manuscript is organized as follows: in Chapter 2 we establish the required NLG, robot-journalism and deep learning background. Chapter 3 describes the methodology we adopted, comparing the most commonly used architectures for automated journalism. Finally, in Chapter 4 we present the results of our work, and in Chapter 5 we conclude our work.

1.3 Published Articles

During the masters program, the author has participated in projects that has led to three papers, which were fundamental for developing this proposal. They are:

- The author published as first author a paper in the "15th International Natural Language Generation Conference" (INLG 2022) with the title "BLAB Reporter: Automated Journalism Covering The Blue Amazon" (SYM; CAMPOS; COZMAN, 2022). This paper was the first step for the present proposal, introducing the initial version of our automated journalism system, which used the pipeline architecture for automated journalism to generate and publish daily reports on Twitter. João Gabriel M. Campos [4] participated in this work as second author, providing guidance on how to design and implement each module of our Blue Amazon agent.

- The author published a paper as co-author in a workshop of the 31st International Joint Conference on Artificial Intelligence and the 25th European Conference on Artificial Intelligence (IJCAI-ECAI 2022) with the title "The BLue Amazon Brain (BLAB): A Modular Architecture of Services about the Brazilian Maritime Territory" (PIROZELLI et al., 2022). Our main contribution was integrating our automated news reporter in a bigger project aimed at creating an artificial agent able to disseminate information about the Brazilian maritime territory. It encapsulates a number of complex and interconnected services concerning ocean knowledge, from

[4] ⟨http://lattes.cnpq.br/6878326093299523⟩

question answering to news reporting, fostering awareness about oceanographic issues — from biodiversity to food supply, from energy resources to climate forecasts.

- The author published a paper in the "19th National Meeting on Artificial and Computational Intelligence" (ENIAC 2022) with the title "Comparing Computational Architectures for Automated Journalism" (SYM et al., 2022). The paper compares different NLG architectures for generating Brazilian Portuguese text using the Blue Amazon as a common domain.

There was significant collaboration with João Gabriel M. Campos, who is also researching automated journalism for natural language generation. His work differs from the present one as he focused on COVID-19 spreading and Legal Amazon deforestation (with public data from DETER, a real-time deforestation satellite monitor). Moreover, he has not focused on comparisons between architectures.

2 BACKGROUND

This chapter concisely reviews some of the main concepts that are necessary for the understanding of this work, as well as the most notable and related literature. The topics covered are Natural Language Generation, Automated Journalism, Transformer Neural Networks, Evaluation Metrics, and Data Augmentation.

2.1 Natural Language Generation

Natural language generation (NLG) is characterized as "the subfield of artificial intelligence and computational linguistics that is concerned with the construction of computer systems than can produce understandable texts in any human language from some underlying non-linguistic representation of information" and, in simple terms, it can be understood as a process which maps from some input data to an output text (REITER; DALE, 2000). NLG algorithms often combine rule-based and machine learning techniques to transform structured data into human-readable language. These algorithms often rely on both Natural Language Processing (NLP) and Natural Language Understanding (NLU), as well as computational linguistics, to autonomously transform structured data into human-readable, grammatically correct, coherent, and contextually appropriate text, as shown in Figure 4. In its essence, NLG aims to automatically generates narratives that describe, summarize or explain input structured data in a human-like manner, taking into account multiple aspects of language, including its structure, grammar, word usage and perception (MELLISH; REITER; LEVINE, 1996). NLG has many benefits, such as reducing the time and cost of producing written content, increasing the speed and accuracy of content production, and improving the scalability of content generation.

In 1997, Ehud Reiter and Robert Dale published the article "Building applied natural language generation systems" (REITER; DALE, 2000), a milestone in Natural Language Generation. The authors give an overview of NLG from an applied system-building perspective, providing suggestions for carrying out requirements analyses and a detailed de-

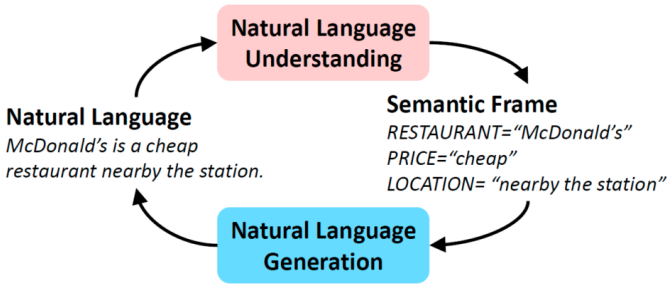

Figure 4: Natural Language Understanding vs. Natural Language Generation. Image obtained from: https://path.com.br/noticias/o-que-e-natural-language-processing.

scription of the most common NLG tasks. The authors also discuss when NLG technology is likely to be appropriate, and when alternative or simpler techniques may provide a more appropriate solution. For instance, in some circumstances, it may be preferable to display the information in the form of charts, schematic diagrams, or schematic diagrams rather than textually. In other situations, text is the most effective form of presentation, although solutions based on the straightforward mail-merge features included in most word processors may be successful, removing the need for more sophisticated NLG approaches. In other situations, hiring someone to develop manuals or to provide user instructions is the best course of action. Typically, the economic choice will be mostly based on the amount of text produced. A NLG system that generates thousands of reports monthly will be easier to defend as cost-effective than one that only generates a few hundred pages annually.

The construction of an initial corpus of human-authored texts and, whenever necessary, their related inputs, is the first stage in a corpus-based requirements analysis. For instance, a business letter corpus may be based on actual letters that were once sent, while a weather report corpus may be based on actual reports that were once produced. The corpus should, to the greatest possible extent, include both common and uncommon examples in addition to the boundary and rare cases that are anticipated to be produced by the NLG system. If there are no human-authored examples of the desired texts, the ideal approach often consists of asking domain experts for insights.

Most NLG applications follow a pipeline approach that applies six sequential steps in order to move from input data to a final output text, as shown in Figure 5: Content Determination (the process of deciding which information should be communicated in the text), Discourse Planning (responsible for imposing ordering and structure over the set of messages to be conveyed), Sentence Aggregation (the process of grouping messages

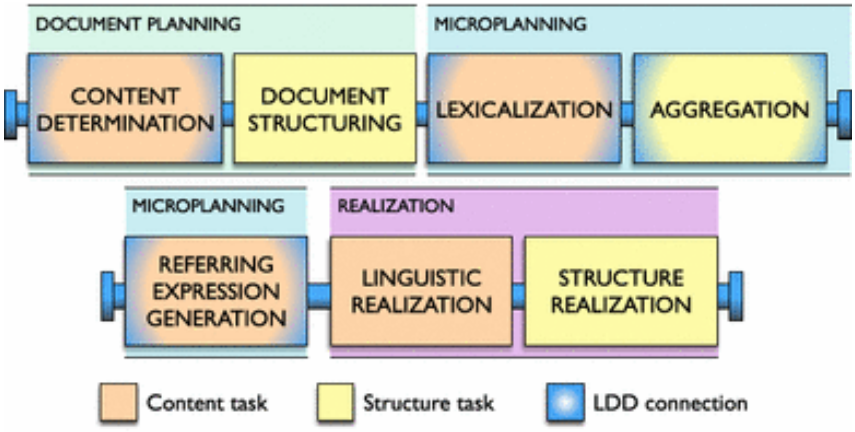

Figure 5: Steps of the Pipeline NLG approach, extracted from (RAMOS-SOTO; BUGARÍN; BARRO, 2016).

together into sentences), Lexicalization (this module decides which specific words and phrases should be chosen to express the domain concepts), Referring Expression Generation (the task of selecting words or phrases to represent nouns, definite noun phrases, spatial and temporal references), and finally the Linguistic Realisation (the process of applying grammar rules in order to produce a text which is syntactically, morphologically, and orthographically correct). These modules will be discussed further in Chapter 3.

As in other fields of Artificial Intelligence, NLG has seen many statistical and neural approaches over the last few years, with the introduction of neural end-to-end methods (ARUN et al., 2020). In such strategies, it is essential to define an adequate corpus, which will be used as a set of training for computer implementations. A promising approach is to use structured data from the Semantic Web to generate a linguistic corpus as a dataset for training text generation implementations, as is the case of WebNLG, created from DBpedia data (COLIN et al., 2016). The DBpedia is a community-wide project that seeks to extract structured information from the contents of Wikipedia pages (AUER et al., 2007). The result of this initiative is the creation of a relational database that, in addition to a vast number of entities, has multilingual support. This data is stored as triples RDF in the form «subject, property, object» where the subject is a URI (Universal Resource Identifier), the property is a binary relation and the object can be another URI, a numeric value or some text type.

NLG techniques are already used in a wide variety of business tools, and they can be seen on a day-to-day basis in, for example, weather forecasting (SRIPADA et al., 2014), customer services, sports reporting in the news (FERNANDES, 2021), and search engines (BHANDARI; BANSAL, 2018). The first attempt to producing commercially feasible NLG applications happened during the 1990s, most notably in the context of

custom letter generation. Stephen Springer and his team at Cognitive Systems in the USA, and José Coch at GSI-Erli in France, developed NLG systems with the goal of automating the creation of fluent customer support responses. For humans, this process took an enormous amount of time and varied greatly in output quality, but it was quite simple for an automated instrument (DALE, 2020).

At the present moment, there are many companies commercially resorting to NLG. Wordsmith is an NLG platform developed by Automated Insights that allows users to create custom templates, reports and summaries. Arria NLG, founded 2013 in the United Kingdom, is believed to be one of the global leaders in NLG technologies and tools, and has patented many commercial NLG technologies in different domains, such as weather forecasting to investment analysis (DÖRR, 2015). The german company AX Semantics, which was founded in 2001, offers e-commerce, journalistic and data reporting NLG services for over a hundred languages, allowing users to create written content from data and helping content creators to scale their writing. Graefe et al. used the software to study readers perception of automated news, coming to the conclusion that the results provide conservative estimates for the favorability of computer-written news (GRAEFE et al., 2018). AX Semantics' technology was also used to power automated election reporting on the 2016 US presidential election with PollyVote (GRAEFE, 2017). Over the course of the project, nearly 22,000 automated news articles were published in English and German. Finally, Yseop is an NLG platform that specializes in generating complex financial and business reports from data.

Modern platforms for natural language production often follow four sequential steps: The software begins by gathering any information that is already available, such as—in the case of baseball—box scores, minute-by-minute plays, batting averages, previous stats, or player demographics (BALDWIN; CHANNARUKUL, 2015). In the second step, algorithms use statistical techniques to find significant and intriguing data points. They could be extraordinary occurrences, a player's outstanding performance, or the turning point in a game. The software next arranges the newsworthy components in accordance with specified rules to create a narrative after classifying and ranking the identified insights according to relevance. Finally, the article can then be uploaded to the publisher's content management system, where it may be automatically published.

The system often uses predetermined rules that are specific to the task at hand and are typically derived by collaborations between engineers, journalists, and computer linguists during this process. For instance, the software must understand that in the game of baseball, the side with the most runs —and not necessarily the most hits— wins. Also, in

order for the algorithm to find interesting events and rank them by importance, criteria for newsworthiness must be defined by domain experts. The last step is to translate the underlying, semantic logic of sample texts into a rule-based system that can create sentences by using computer linguists. If no such examples of texts are available, trained journalists pre-write text modules and examples of stories using the proper language and frame structures, then conform them to the publishing outlet's official style guidelines.

The ability to understand data and to make judgments, such as determining whether a particular pattern of data should be summarized or not, is one of the most difficult tasks for NLG. For example, determining whether an episode of bradycardia is temporary or prolonged without any additional context is far from current art (UCHENDU et al., 2021). Although NLG systems often pass international evaluation contests that focus on one or more particular aspects of language use, many authors have claimed that it will still take many decades of research before NLG programs are able to consistently addressing a Turing Test (FRENCH, 2000). However, with the recent introduction of ChatGPT (Generative Pre-trained Transformer) in November 2022, this might happen sooner than expected (HAQUE et al., 2022). ChatGPT is a large language model chatbot developed by OpenAI based on GPT-3.5 family of large language models, and is fine-tuned using both supervised and reinforcement learning methods (KAELBLING; LITTMAN; MOORE, 1996). It is capable of engaging in conversational dialogues and responding in ways that seems surprisingly human, and is already a core part of Microsoft's Bing. While the core function of a chatbot is to mimic human communication, ChatGPT also has versatility and improvisation skills, being also able to write poetry, compose music, write computer programs, answer questions and play games like tic-tac-toe (CASTELVECCHI, 2022). However, ChatGPT sometimes writes plausible-sounding but incorrect or nonsensical answers. This behavior often arises in large language models and is called artificial intelligence hallucination (RUDOLPH; TAN; TAN, 2023). Google has developed its own large language model, called Bard, as a direct response to the rise of OpenAI's ChatGPT.

During 2023, NLG systems based on ChatGPT have been applied in many areas, such as healthcare, finances, and e-commerce. Such systems help content producers generating product descriptions, reviews, and other marketing content for websites and digital platforms. They are also used to improve relevance and personalization of product recommendations and to increase customer engagement and satisfaction.

2.2 Automated Journalism

In automated journalism, also known as algorithmic journalism or robot journalism, news articles are generated in human-readable ways by computer programs using NLG techniques. This often entails storing and scanning huge amounts of provided data for a particular domain, choosing from a variety of NLG approaches and pre-programmed layouts, arranging key points, and inserting details such as names, places, numbers, ranks, statistics, and other relevant figures. Additionally, the output may be altered to match a certain tone, voice, style, or even a specific audience (MONTAL; REICH, 2017).

Automated journalism is often viewed as a opportunity to relieve journalists of ordinary reporting and provide them more time for difficult tasks, such as investigative reporting and in-depth event analysis (TOUSIGNANT, 2020), and is gaining popularity among news organizations because it can quickly produce large amounts of news stories with minimal human intervention.

In its simplest form, text can generated by keeping a list of possible texts and filling the gaps with previously collected information. Applications that utilize this method generate text by directly mapping non-linguistic input (i.e., without intermediate representations, such as tabular data) to a linguistic structure. The linguistic structure is generated by filling in gaps according to context and manipulating strings. The main benefit of this approach is the simplicity of the implementation, and the main negative point is the difficulty of representing the text in any deeper way, as there is no planning for the discourse and no lexical variability (DEEMTER; THEUNE; KRAHMER, 2005). The results using this approach may be satisfying in less complex domains, such as weather forecasting, horoscope machines or generating personalized business letters (DEEMTER; THEUNE; KRAHMER, 2005).

Recent years have seen a paradigm shift in NLG with the emergence of deep contextual language modeling (e.g., LSTMs, transformers networks) and transfer learning (e.g., T5, BERT, GPT). This is called the neural end-to-end approach; it handles non-linguistic input in natural language without explicit intermediary representations. Although these tools have significantly improved automated journalism, state-of-the-art NLG models still face a number of challenges, including commonsense violations in depicted situations, challenges utilizing factual information, and challenges creating trustworthy evaluation metrics (CHEN et al., 2019). Three commonly used architectures for automated journalism are: template, pipeline and neural end-to-end, and they will be further discussed in Chapter 4. Figure 6 shows the business value created by NLG systems.

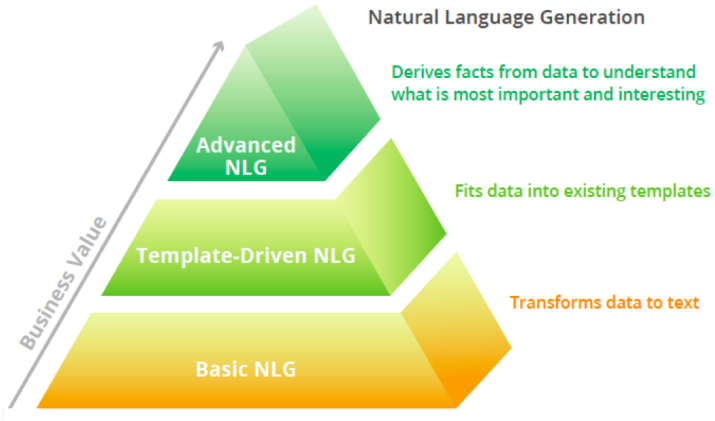

Figure 6: Business value created by NLG systems. Image obtained from: https://medium.com/@narrativesci/defined-natural-language-generation-22c28c3524e5

.

According to Francesco Marconi of the Associated Press, automation allowed the news organization to free up 20% of reporters' time so they could work on initiatives with more impact (LINDÉN et al., 2019). Because more content can be created faster, automated journalism is less expensive and relieves human journalists from repetitive work, allowing them to focus on more significant news.

Early implementations of automated journalism started in the 1990s; they were used primarily to tell stories using numerical and statistical data. Now, data science and AI companies such as Automated Insights, Narrative Science, United Robots and Monok develop and provide automated journalism algorithms for many use cases, such as financial reporting, sports recaps and chatbots (DÖRR, 2015). StatSheet, a website that follows college basketball, runs entirely based on robot-journalism algorithms. More famously, an algorithm called Quakebot published a story about a 2014 California earthquake on The Los Angeles Times website within three minutes after the shaking had stopped (DÖRR, 2015). The Los Angeles Times' Quakebot exhibits the use of sensor data for automated journalism by automatically writing brief news pieces about earthquakes in California. When the U.S. Geological Survey's Earthquake Notification Service issues an earthquake alert, Quakebot creates a story and saves it as a draft in the content management system of the Los Angeles Times that includes all the essential details a journalist would initially cover, such as the time, location, and magnitude of the earthquake. After a staff member reviews the story for potential errors, it only takes a single click to publish the story.

The key to Quakebot is speed, and its objective is to disseminate knowledge as soon as possible. While accuracy of the news is just as crucial as speed, accomplishing both objectives can be challenging. The quality of the underlying data is a critical component

Figure 7: Los Angeles Times apologized for mistakenly reporting an earthquake in Isla Vista.

of accuracy for automated news. This became evident in May 2015 when seismologic sensors in Northern California picked up signals from major earthquakes that happened in Japan and Alaska, which the U.S. Geological Survey (USGS) mistakenly reported as three separate earthquakes in California with magnitudes ranging from 4.8 to 5.5. Earthquakes of that magnitude would leave significant local damage, but the alarms turned out to be false. The earthquakes never happened and nobody could feel them, but Quakebot wrote articles about each of the three false alarms. Another mistake happened in 2017, when Quakebot mistakenly reported an earthquake in Isla Vista, but it had happened in 1925. Los Angeles Times made a public tweet apologizing for the mistake, as shown in Figure 7. Without verifying that the information was accurate, the editor believed the algorithm and published the piece.

As the technology for automated journalism continues to develop, it is likely to become more sophisticated and capable of producing higher-quality content. However, it is also important to ensure that the journalistic standards of accuracy, fairness, and transparency are maintained. Recent studies in which humans had to evaluate the reliability, value, and readability of automated news compared to human-written news found no relevant differences in readers' perceptions of credibility, a slight advantage for human-written news in terms of quality, and a significant advantage for human-written news in terms of readability (GRAEFE; BOHLKEN, 2020). Experimental comparisons also show that when participants were told they were reading a human-written piece, they gave the article better scores for quality, credibility, and readability. These results highlight the ethical issues that arise from automated journalism and may cause news companies to withhold the information that a report was generated automatically. As with any form of journalism, automated journalism must adhere to ethical principles to ensure the accuracy, fairness, and transparency of the content produced (TORRIJOS, 2021). Key ethical considerations for automated journalism are:

- **Accuracy:** Automated journalism should always strive for accuracy in reporting. Algorithms and software can make mistakes, and errors in news reporting can have serious consequences. Automated journalism systems should be programmed to fact-check and verify the information before generating content. News organizations should also have a process for reviewing and correcting errors that may occur.

- **Transparency:** News organizations should be transparent about their use of automated journalism. Readers should be aware that the content they are reading was generated by an algorithm, not a human journalist. There are ongoing discussions whether organizations should also provide information regarding how their algorithms work and what sources of data they use.

- **Fairness:** Automated journalism should not perpetuate biases or discrimination. News organizations should be aware of the potential biases in the data used to train the algorithms and take steps to mitigate those biases. They should also ensure that the content produced is balanced and free from discrimination.

- **Accountability:** News organizations should take responsibility for the content produced by automated journalism systems. They should have processes in place to monitor the content and ensure that it meets ethical standards. They should also be prepared to take corrective action if issues arise.

- **Human oversight:** While automated journalism can reduce costs and increase production speed, it should not replace human journalists entirely. There should be human specialists checking the generated news to ensure that the content produced is accurate, fair, and ethical.

- **Intellectual property:** Automated journalism systems can generate content that is protected by copyright and other forms of intellectual property, being often perceived as a threat to the authorship. It is important to ensure that automated journalism systems are designed to respect these rights and that they do not infringe on the intellectual property of others.

The accountability and transparency of algorithms are two more crucial challenges for the use of automated journalism in newsrooms. There is insufficient information about, for example, whether news consumers need to comprehend how algorithms work or what data they utilize to produce content. Furthermore, little information exists regarding the data that news companies should make transparent and the inner workings of their

algorithms, aside from a few fundamental rules and criteria that should be adhered to when using automation technology.

In 2006, Anja Belz and Ehud Reiter published the article "Comparing Automatic and Human Evaluation of NLG Systems" which set the benchmark for evaluating NLG systems (BELZ; REITER, 2006). They presented empirical studies on how well various corpus-based metrics agree with human judgments for evaluating several NLG systems. Their paper utilizes automatic evaluation metrics, such as BLEU and ROUGE, which are used in our work, and compares them with human evaluations on empirical cases. They also compared the results obtained by human domain experts and by human non-experts.

Traditionally, most NLG systems have been evaluated using human subjects (MELLISH; DALE, 1998). Human evaluation metrics are often divided into intrinsic, where subjects are shown both NLG and human-written texts and the NLG system is evaluated by comparing their ratings, and extrinsic, which measure other aspects such as how quickly the texts can be read, their impact on a given task performance and how many edits experts need to make to correct the texts. Corpus-based evaluation for NLG was first introduced in 1998, where texts were parsed from a corpus and the output was fed to the NLG system, with the goal to compare the generated texts to the original texts (LANGKILDE; KNIGHT, 1998). Such corpus-based evaluations have sometimes been criticised in the NLG community (REITER; SRIPADA, 2002), because regenerating a parsed text is not a realistic NLG task, and the texts can be very different from the original text but still effectively communicate the intended message.

The authors conclude that automatic evaluation metrics for NLG systems are very promising, mainly in contexts where there are high-quality reference texts and only a small number of human evaluators are available. However, in general the authors recommend that it is best to combine automatic metrics with human-based evaluations in scenarios where it is unknown whether they are correlated. They also found out that NIST (DODDINGTON, 2002) is a more reliable evaluation metric than BLEU and ROUGE, and that individual experts' judgments are not likely to correlate highly with average expert opinion.

In 2019, Ferreira et al. published the article "Neural data-to-text generation: A comparison between pipeline and end-to-end architectures" which served as an inspiration for our work by introducing a systematic way to compare different approaches for generating text from structured input data (FERREIRA et al., 2019). The authors used the WebNLG corpus to conduct their experiments, which consists of sets of <Subject,

Predicate, Object> RDF triples and their target texts in English (GARDENT et al., 2017).

The authors carried out automatic and human evaluations together with a qualitative analysis and concluded that having explicit intermediate steps in the generation process results in better texts than the ones generated by end-to-end approaches. While cascading of errors is a problem of pipeline models in general (an error in an early module will impact all later modules in the pipeline), developing dedicated neural modules for specific tasks leads to better performance on each of these successive tasks, and combining them leads to better, and more reusable, output results. The texts were evaluated using the BLEU and METEOR metrics, as well as human evaluation metrics. For each sample, the authors used the original texts and the ones generated by their NLG approaches, for a total of 2,007 trials. Each trial displayed the triple set and the respective text. The goal of the participants was to rate the trials based on the fluency and semantics in a 1-7 Likert scale. We adopted a similar metric in our work, but we opted to use a 1-5 Likert scale.

One of the most prominent NLG systems in Brazilian Portuguese is the DaMata Reporter (Figure 8), a robot-journalism system developed in 2020 to cover deforestation in the Brazilian Amazon (TEIXEIRA et al., 2020). The application generates multilingual daily and monthly reports based on the public data provided by DETER, a real-time deforestation satellite monitor developed and maintained by the Brazilian National Institute for Space Research (INPE), and publishes them on Twitter. The system follows a pipeline architecture instead of the novel end-to-end systems in order not to hallucinate content, which is very problematic in this particular domain. The system converts non-linguistic data into text in six steps: Content Selection, Discourse Ordering, Text Structuring, Lexicalization, Referring Expression Generation and Textual Realization.

The grammar used by DaMata was built by running the content selection step in previous collected data, generating 14 non-linguistic monthly reports and 25 daily ones. These reports were then manually verbalized and the input and output representations for each pipeline module were manually annotated, resulting in a list of possible discourse orders, text structures, lexicalizations, and referring expressions. When deployed, each module draws on the selected combination of templates using a list of rules developed by the authors.

Another automated journalism system in Brazilian Portuguese is Rui Barbot (Figure 9), developed by Jota, an automated journalism agent specialized in covering Brazilian justice. Every time it notices a legal case at the Brazilian Supreme Court halted for more

Figure 8: DaMata Reporter: an automated journalism agent specialized in covering the deforestation in Brazil.

than 180 days, Rui Barbot tweets about it. The application keeps track of lawsuits in the Brazilian Supreme Court (STF), and when Rui notices that a case hasn't had any fresh developments for longer than six months, it tweets an alert. The system uses a simple template architecture to communicate with the audience, and doesn't provide for lexical variety. Templates are pre-written sentences or paragraphs that contain placeholders for variable information. These placeholders are then replaced with specific data to generate text.

2.3 Transformer Neural Networks

Here we will discuss transformer neural networks by addressing their theory and implementation.

2.3.1 Theory

Transformer neural networks are deep learning models that were introduced in 2017 and aim to solve sequence-to-sequence tasks while handling long-range dependencies (VASWANI et al., 2017b). They utilize the mechanism of self-attention to weigh the significance of each part of the input data, and were originally designed for NLP tasks, such as machine translation and language generation, but have since been applied to other areas such as speech recognition and computer vision. The core idea behind the transformer architecture is the use of self-attention mechanisms to allow the model to

Figure 9: Rui Barbot: an automated journalism agent specialized in covering Brazilian justice.

selectively focus on different parts of the input sequence when generating the output. This is achieved through a "multi-head self-attention mechanism" that allows the model to attend to multiple parts of the input sequence simultaneously.

The transformer architecture also includes a feedforward neural network that is applied to each position in the sequence independently, as well as layer normalization and residual connections to improve training stability. The transformer architecture has been widely adopted in the field of NLP, and has achieved state-of-the-art results on a variety of tasks, including machine translation, language modeling, and sentiment analysis. An example of a transformer neural network architecture is shown in Figure 10.

Transfomer neural networks are often utilized with transfer learning, which is the process of applying knowledge gained while solving one problem to a different but related problem (WEISS; KHOSHGOFTAAR; WANG, 2016). For example, knowledge gained while learning how to communicate with a human on a finance context can be used in a retail context, and knowledge obtained by learning how to recognize dogs can help when trying to recognize cats by fine-tuning to the specific tasks. Pre-trained models are frequently used due to the enormous time and computing resources needed to develop neural network models for these problems as well as the enormous skill gains they offer on similar contexts. In computer vision, for example, neural networks often attempt to detect edges in the earlier layers, shapes in the middle layer, and some task-specific properties in the latter layers in computer vision. The early and intermediate layers are utilized in transfer learning, and the latter layers are only retrained. This aids in utilizing the labeled data from the initial task it was trained on.

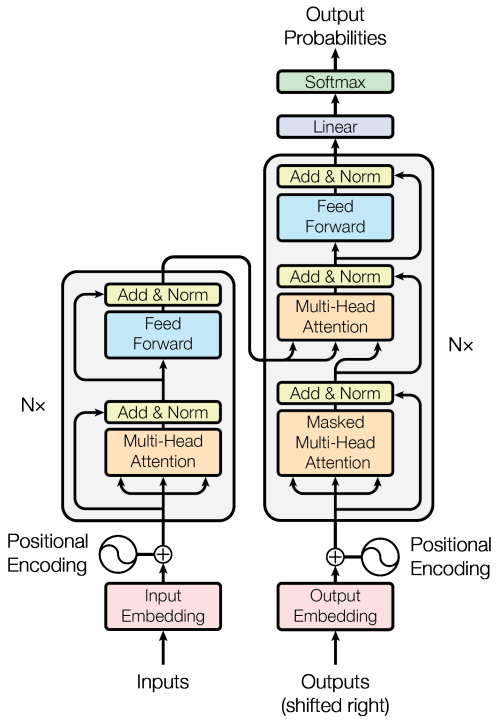

Figure 10: Example of a transformer neural network architecture (VASWANI et al., 2017a).

One of the most common pre-training techniques for transformer networks in NLG is Masked-Language Modeling, which is done using many texts of different kinds: in each excerpt, one or more words are masked, and the network is trained to predict the original sentence. For example: for the sentence "You must be the [MASK] you wish to see in the world" (Mahatma Ghandi), the network has to complete it with the word that makes the most sense in this context (the correct word is "change").

There are several pre-trained transformers networks, but we will just talk about the four that apply to this work:

- **BERT:** Bidirectional Encoder Representations from Transformers (BERT) is one of the most well-known pre-trained transformer neural networks. Unlike other encoder networks that are only able to read input sequentially, BERT reads the complete input sequence all at once (DEVLIN et al., 2018). This makes it possible to better understand the context involved in the sentence, making it bidirectional.

- **T5:** Text-To-Text Transfer Transformer (T5) is an encoder-decoder transformer network that has been pre-trained to perform a variety of tasks, some of which are comparable to BERT, but each of which is then converted into a text-to-text arrangement (XUE et al., 2020). By simply changing the prefix, it can be used for various tasks with the same network, such as machine translation and summariza-

tion.

- **Blenderbot**: Blenderbot is an encoder-decoder transformer network which has been pre-trained to generate responses, interacting and responding like a conversational agent (SHUSTER et al., 2022). It can build long term memory for continuous access while simultaneously searching the internet for up-to-date information, holding conversations on nearly any topic.

- **GPT2:** The second generation Generative Pre-trained Transformer (GPT2) is a unidirectional decoder network capable of generating texts for a variety of purposes (VIG; BELINKOV, 2019). It is a model that is comparable to the original GPT network, but with roughly ten times as many parameters and a significantly larger training dataset.

2.3.2 Implementation

The most frequently used Python library for working with transformers is the "Transformers" library maintained by the HuggingFace team [1]. This library provides APIs to download most state-of-the-art transformers models and provides useful tools for training neural networks. HuggingFace supports all of the models used in this work.

2.4 Evaluation Metrics

Simple NLP tasks such as sentiment analysis are often easy to evaluate since the evaluation simply requires label matching (NASUKAWA; YI, 2003). As such, metrics like F-score (which is the harmonic mean of precision and recall), or even accuracy in uniformly distributed data, are used for such tasks (LIPTON; ELKAN; NARAYANASWAMY, 2014). However, evaluating natural language generation systems is a much more complex task. And for this reason, a number of different metrics have been proposed for tasks such as machine translation and text summarization (FRISONI et al., 2022).

Evaluation methods are often divided into two categories: a comparison to gold reference, or an appeal to human judgement. Automated assessment techniques which compare generated output to a gold reference sometimes combine fidelity and fluency into a single output score (SAI; MOHANKUMAR; KHAPRA, 2022). BLEU (PAPINENI et al., 2002) is a canonical example: the metric assesses faithfulness by counting the matches between

[1] https://huggingface.co/docs/transformers/index

n-grams in a candidate translation text and those in a reference text, and evaluates fluency by implicitly employing the reference n-grams as a language model. Several human judgement frameworks specifically request distinct judgments on task elements that correspond to fidelity and fluency because we frequently want to know the quality of the two qualities independently. Also, it can be difficult to characterize a feature of text quality that is not connected to the language used to express the text's meaning but rather to the assessment measure without using reference texts, which intuitively seem unneeded.

This work employs three traditionally used automatic evaluation metrics for NLG: BLEU, ROUGE (LIN; OCH, 2004), and METEOR (LAVIE; DENKOWSKI, 2009), as well as four more recent purposed metrics: GLEU (MUTTON et al., 2007), BLEURT (SELLAM; DAS; PARIKH, 2020), COMET (FEINER; MCKEOWN, 1991), and PRISM (THOMPSON; POST, 2020). These are the most commonly used approaches for evaluating the quality of generated text from one natural language to another. Quality is taken to be the correspondence between a machine's output text and that of a human: the closer a machine generated text is to a human-written reference text, the better it is (CASACUBERTA et al., 2009).

- **BLEU:** Bilingual Evaluation Understudy (BLEU) is the most popular metric for evaluation the quality of generated texts. Quality is considered to be the correspondence between a machine's output and that of a human: "the closer a machine translation is to a professional human translation, the better it is" (REITER, 2018). BLEU was one of the first metrics to claim a high correlation with human judgements of quality, and remains one of the most popular automated and inexpensive metrics. BLEU's output is always a number between 0 and 1, with values closer to 1 representing more similar texts. It has been argued that although this metric has significant advantages for general use cases, there is no guarantee that an increase in the BLEU score is an indicator of improved translation quality, since other factors such as intelligibility and grammatical correctness are not taken into account. The BLEU metric is defined as:

$$\text{BLEU} = \text{BP} \times \exp\left(\sum_{n=1}^{N} w_n \log(p_n)\right),$$

  where:

  - BP is the brevity penalty;
  - N is the maximum n-gram order to consider;

  - w_n is the weight assigned to the n-gram precision score;
  - p_n is the n-gram precision score.

- **ROUGE:** Recall-Oriented Understudy for Gisting Evaluation (ROUGE) is a set of metrics used for evaluating automatic summarization and machine translation systems. The metrics compare an automatically produced summary or translation against a reference or a set of references (human-produced) summary or translation. ROUGE is primarily based on the concept of n-gram overlap, where n-grams are contiguous sequences of words of length n. There are several variants of the ROUGE metric, each using a different type of n-gram overlap:

  ROUGE-1 measures the overlap of unigrams (single words) between the generated text and the reference text. ROUGE-2 measures the overlap of bigrams (pairs of adjacent words) between the generated text and the reference text. ROUGE-N is a variant of the ROUGE metric that focuses on n-gram overlap between the generated text and the reference text. In ROUGE-N, the "N" refers to the length of the n-grams being considered. ROUGE-L measures the longest common subsequence (LCS) between the generated text and the reference text, which is a sequence of words that appear in the same order in both texts. This metric is particularly useful in cases where summaries may contain paraphrases or rephrasings of the same information, as it captures the similarity of the information rather than just the exact words.

  ROUGE-W and ROUGE-S are based on the weighted longest common subsequence (LCS), and skip-bigram co-occurence statistics, respectively (ZHANG et al., 2018). Instead of using only recall, these use an F-score which is the harmonic mean of precision and recall values. For our work, we utilized ROUGE-1, which counts recall based on matching unigrams. ROUGE-N can be calculated as:

$$ROUGE\text{-}N = \frac{\sum_{\text{n-gram} \in \text{machine-generated summary}} \text{countmatch(n-gram)}}{\sum_{\text{n-gram} \in \text{reference summary}} \text{count(n-gram)}}.$$

- **METEOR:** Metric for Evaluation of Translation with Explicit ORdering (METEOR) is a metric based on the harmonic mean of unigram precision and recall, with recall weighted higher than precision. It claims to have better correlation with human judgement, presenting several features that are not found in other evaluation metrics. METEOR also modifies the precision and recall computations, replacing them with a weighted F-score based on mapping unigrams and a penalty function

for incorrect word order. The metric also uses more advanced techniques, such as stemming and synonymy matching, along with the standard exact word matching. Overall, METEOR is a complex metric that takes into account both the quality of the alignment between the generated text and the reference text, as well as the fluency and grammaticality of the generated text. It has been shown to be effective in a wide range of NLG tasks, including machine translation, summarization, and text generation. It is defined as:

$$\text{METEOR} = (1 - \gamma) \times P_r \times R \times F_{mean} + \gamma \times \Delta,$$

where:

- γ is a tunable parameter that controls the balance between precision and recall;
- P_r is the paraphrase score;
- R is the unigram recall score;
- F_{mean} is the mean of the harmonic mean and the arithmetic mean of the unigram precision and recall scores;
- Δ is the penalty term for shifts in word order or function words.

- **GLEU:** Automatic Evaluation of Sentence-Level Fluency (GLEU) is a metric purposed by Google which applies the minimum of the precision and recall. It is a modified version of the BLEU metric that incorporates the concept of gap-weighting, which penalizes the model more for larger gaps in the generated sentence. The GLEU score has a high correlation with the BLEU score on a corpus level but does not have its drawbacks for the per sentence reward objective. While the majority of the automatic evaluation techniques combines measurement of faithfulness to a given source content with fluency of the resulting text, GLEU estimates fluency alone by examining the use of parser outputs as metrics. GLEU is defined as:

$$GLEU = \min(1, e^{1-(reference\_length/output\_length)} \frac{matched\_ngrams}{output\_ngrams}^{1/n}), \qquad (2.1)$$

where the variables have the same meanings as described previously. Note that this equation assumes that the variables have already been preprocessed and calculated, and the result is a single value between 0 and 1.

- **BLEURT:** Bilingual Evaluation Understudy with Representations from Transformers (BLEURT) builds upon recent advances in transfer learning to capture

widespread linguistic phenomena, such as paraphrasing. The BLEURT score is based on the BLEU score, but is computed using a pre-trained transformer model, which allows it to take into account the meaning of the words and the context in which they are used. Popular artificial measures (like BLEU) are frequently unreliable replacements for human interpretation and judgment, and the need for new methods of assessment is prompted by the rapid advancement of NLG and the shortcomings of current evaluation techniques. We later use BLEURT-20, which was trained on multiple languages, including portuguese. The BLEURT metric can be calculated as:

$$\text{BLEU} = \text{BP} \times \exp\left(\sum_{n=1}^{N} w_n \log p_n\right),$$

where:

- BP is the brevity penalty, which is 1 if the candidate translation is longer than the reference translation and is computed as $\exp(1 - r/c)$ if the candidate translation is shorter or equal to the reference translation, where r is the length of the candidate translation and c is the length of the closest reference translation;
- w_n is the weight assigned to the n-gram precision score, which is typically set to $\frac{1}{N}$ for equal weighting;
- p_n is the n-gram precision score, which is computed as the number of n-grams in the candidate translation that match exactly with n-grams in the reference translation, divided by the total number of n-grams in the candidate translation.

- **COMET:** Crosslingual Optimized Metric for Evaluation of Translation (COMET) is an open-source neural framework used to train multilingual machine translation evaluation models which achieves new state-of-the-art levels of correlation with human judgements. It fine-tunes a multilingual language model named XLM-R (CONNEAU et al., 2020) to predict human translation texts.

- **PRISM:** Probability is the metric (PRISM) is an unsupervised text generation metric which scores text outputs based on their corresponding human references using a sequence-to-sequence zero-shot paraphrase model, estimating the probability that an output is a paraphrase of the reference text. This process eliminates the requirement for synthetic paraphrase data and produces a single model that is

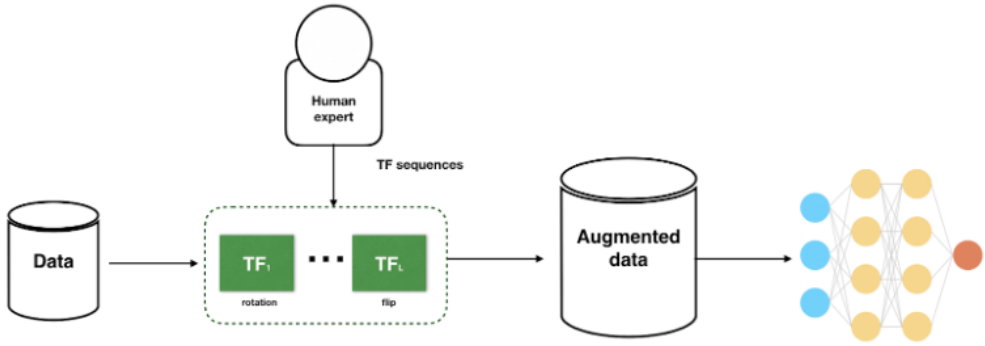

Figure 11: Data augmentation for image related machine learning tasks (extracted from: http://ai.stanford.edu/blog/data-augmentation).

functional across numerous languages. To score a text, the reference is supplied into the encoder, and the decoder force-decodes the hypothesis. The score is created by aggregating the token-level probabilities of the reference, and then repeating the procedure with the reference in the decoder and the hypothesis in the encoder. The two scores are averaged to produce the final score.

2.5 Data Augmentation

Data augmentation is a technique where the training set is augmented by generating slightly modified copies of existing data or newly created synthetic data. It is a process closely related to oversampling in data analysis, and it aims to expand the size and diversity of the training set, acting as a regularizer and helping to reduce the probability of both model underfitting (DYK; MENG, 2001). The goal of data augmentation is to create a more diverse and representative dataset that can improve the accuracy and generalization of machine learning models. By generating new examples that are similar to existing ones, but with some variations or modifications, data augmentation can help to improve the robustness and performance of a model (TANNER; WONG, 1987). Figure 11 illustrates the complete pipeline for training neural models using data augmentation.

For image related machine learning tasks, if a dataset is relatively small, adding rotation, mirroring, zooming, grayscaling, and filtering can be used as data augmentation techniques. If they are still not sufficient to solve a particular issue, it is possible to create brand-new, synthetic images using a variety of techniques, such as generative adversarial networks (GANs) and neural style transfer. Figure 12 illustrates some of the most employed data augmentation techniques for dealing with images.

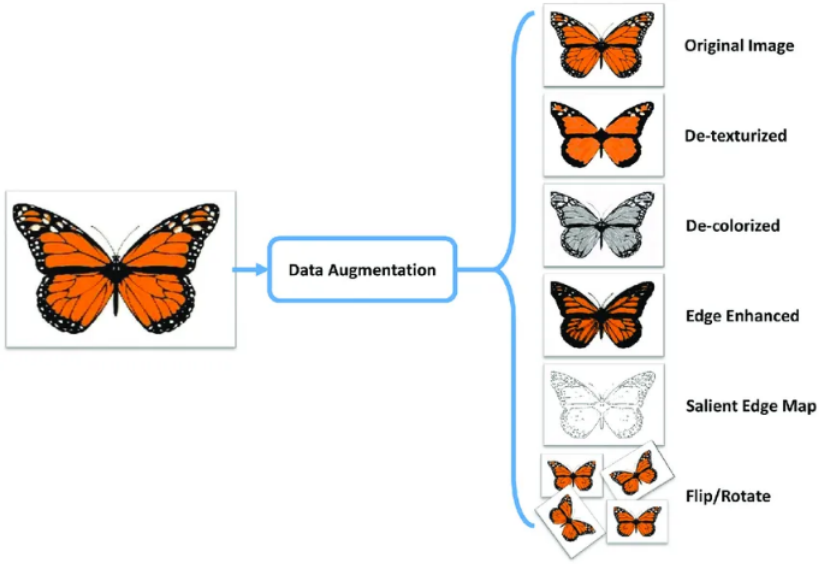

Figure 12: Data augmentation for image related machine learning tasks. https://medium.com/secure-and-private-ai-writing-challenge/data-augmentation-increases-accuracy-of-your-model-but-how-aa1913468722

.

Data augmentation for textual data is typically employed when there is a dearth of high-quality data, and raising performance metrics is the main goal. It is possible to use either character or word swapping, random insertion or deletion, word embeddings, and synonym augmentation. These methods are specially helpful for languages with limited resources or in scenarios where few texts are available. For our work, we used number replacements (e.g. swapping the temperature value from 30 to 31), and synonym augmentation to generate more Blue Amazon data.

The key benefits of data augmentation include:

- Improving model prediction accuracy.
- Reducing data overfitting by increasing the generalization ability of the models.
- Helping resolve class imbalance issues.
- Reducing the costs of collecting and labeling new data.

It is worth noting that data augmentation should be used with care, as some transformations or manipulations of the data may not be appropriate or may even introduce bias into the dataset. Additionally, the effectiveness of data augmentation depends on the quality and diversity of the original dataset, as well as the specific problem and task being addressed.

3 METHODS: ARCHITECTURES AND DATA

In this work, we studied and applied the three most commonly used architectures for automated journalism: template, pipeline and neural end-to-end. We describe the infrastructure we built to host the automated journalism agent, the data we collected and used for the experiments, and the systems we have built.

3.1 Blue Amazon Data

The experiments presented in this work were run with information from the Blue Amazon domain. Data was collected during a period of 3 consecutive months from publicly available sources, including weather, temperature, tides charts, earthquakes, vessel positioning and oil extraction. All the sources of information were validated by two phd researchers from the oceanography course at the University of São Paulo (USP), who we considered to be our domain experts.

The automated journalism agent is hosted in a server and it performs data extraction every four hours using web crawlers. Web crawling is a technique to collect data from the web by finding all the URLs for one or multiple domains, and in this project it was performed using Python together with the Beautiful Soup and Selenium open-source libraries (MIRTAHERI et al., 2014). After the data is collected and cleaned, it is stored in the MongoDB database, a NoSQL document-oriented database program that provides more flexibility and scalability over relational databases when input data is constantly changing (SAHATQIJA et al., 2018). After storing the data, a module of our application is responsible for gathering information from the MongoDB database, merging by city and day, translating textual data to Portuguese and storing it in a tabular format, as shown in Table 1. The tabular data is used as input for our automated journalism agent, which combines different NLG architectures to transform structured data into coherent output text. The resulting text is published on Twitter using the open-source tweepy library. Figure 13 shows the design of our infrastructure for our automated journalism agent.

Table 1: Structured data used as input for the template and pipeline architectures of NLG.

Date	City	UF	Weather	T	T (d-1)	Max T (30d)	Min T (30d)
15/12/2022	Rio de Janeiro	RJ	Sunny	37°C	33°C	True	False
20/02/2023	Salvador	BA	Cloudy	30°C	29°C	False	False
10/03/2023	Recife	PE	Rainy	28°C	27°C	False	False

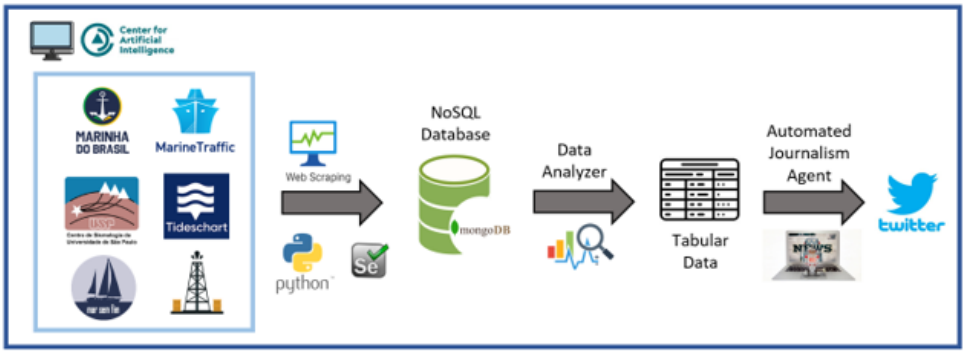

Figure 13: Infrastructure developed for the Blue Amazon automated journalism agent.

The table generated by the analyzer module is used as input for both the template and pipeline architectures. For each line in the table, our system looks up for intents to be verbalized and composes the resulting text to be published as a tweet. For the first line in Table 1, some of the possible intents would be the peak of temperature in the last 30 days, temperature 12% higher than the previous day, and the sunny weather. For training the neural end-to-end models, the tabular data is converted into a intent-attribute-value semantic representation. For example, the first line on Table 1 would become:

```
LOCATION(city="Rio de Janeiro", state="RJ");
WEATHER(climate="Sunny", temperature="37ºC", peak="True", increment="12%")
```

Weather data and tides charts are extracted from the Tides Chart website [1], which provides information about high tides, low tides, tide charts, fishing times, ocean conditions, water temperatures and weather forecasts for thousands of cities around the world. Figure 14 (left) shows an example of tides charts for the coming week in Rio de Janeiro (RJ).

Vessel positioning is collected from the Marine Traffic website [2], which is an open, community-based project that provides real-time information about ship movements around the world and also their current location in ports and harbors. The project was originally developed as an academic project at the University of the Aegean and contains a database

[1] https://www.tideschart.com
[2] https://www.marinetraffic.com

Day	Tide times for Rio de Janeiro					
	1st Tide	2nd Tide	3rd Tide	4th Tide		
Sun 28	3:06am ▲ 1.3 m	9:57am ▼ 0 m	3:43pm ▲ 1.2 m	10:08pm ▼ 0.3 m	▲ 6:07am	▼ 5:41pm
Mon 29	3:38am ▲ 1.3 m	10:21am ▼ 0 m	4:11pm ▲ 1.2 m	10:26pm ▼ 0.3 m	▲ 6:06am	▼ 5:42pm
Tue 30	4:09am ▲ 1.3 m	10:44am ▼ 0.1 m	4:41pm ▲ 1.2 m	10:37pm ▼ 0.4 m	▲ 6:05am	▼ 5:42pm
Wed 31	4:44am ▲ 1.2 m	11:08am ▼ 0.3 m	5:15pm ▲ 1.1 m	10:41pm ▼ 0.4 m	▲ 6:04am	▼ 5:42pm
Thu 1	5:30am ▲ 1.1 m	12:27pm ▼ 0.4 m	6:01pm ▲ 0.9 m	10:49pm ▼ 0.4 m	▲ 6:03am	▼ 5:43pm
Fri 2	6:36am ▲ 1 m	1:48pm ▼ 0.5 m	6:54pm ▲ 0.8 m	11:08pm ▼ 0.4 m	▲ 6:02am	▼ 5:43pm
Sat 3	12:49am ▲ 0.5 m	1:15am ▼ 0.5 m	7:40am ▲ 0.9 m	11:22am ▼ 0.7 m	▲ 6:01am	▼ 5:43pm

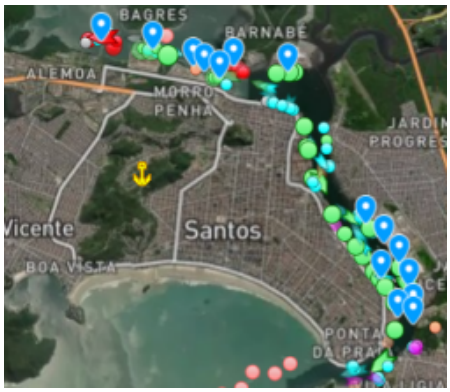

Figure 14: Left: tides chart for the Rio de Janeiro (RJ) city, taken from the Tides Chart website. Right: vessel positions near the Santos (SP) port on a given day, taken from the Marine Traffic website.

Centro de Sismologia USP

Hora de Origem	Long.	Lat.	Prof.	M	Mag.	Região
2022-08-22 14:55:53 UTC	-44.08	-19.57	0.0	M	2.2 mR	Matozinhos/MG
2022-08-19 09:33:16 UTC	-39.86	-9.76	0.0	M	2.5 mR	Uaua/BA
2022-08-14 12:30:36 UTC	-51.65	-15.32	0.0	M	3.3 mR	Araguaiana/MT
2022-08-08 23:09:47 UTC	-35.79	-5.45	0.0	M	2.1 mR	Rg. de João Câmara/RN

Figure 15: Recently detected earthquakes in Brazil. Image obtained from the Seismological Center at the University of São Paulo.

of vessel which uses the Automatic Identification System (AIS). Ship locations are shown on a Google Maps background using the Google Maps API, Nautical Charts and OpenStreetMap. Figure 14 (right) shows an example of vessel positions near the Santos (SP) port on a given day. Finally, real time data regarding earthquakes in the Brazilian coast are taken from the Seismological Center at the University of São Paulo [3]. Figure 15 shows an example of recent earthquakes detected in Brazil.

Based on the work described in (FERREIRA et al., 2019), we created the corpus with information for 100 consecutive days for 50 cities in the Brazilian coast, and then performed content selection for past time-series data using feedback from two domain experts. The intent messages were then sorted using a rule-based approach, and verbalizations of the intent messages were performed by the authors based on a sample of 500 rows from the input table. Syntactic and lexical patterns in the samples were used to produce a variety of target intent texts. Finally, intermediate representations in the

[3]https://www.tideschart.com

pipeline steps were annotated in a intent-attribute-value format and used as input for the neural end-to-end approach. Some examples of intent-attribute-value in the dataset are:

```
LOCATION(city="Santos",uf="SP",timestamp="Jan 15, 2022");
EARTHQUAKE(magnitude="1.4 mR",depth="15km");
WEATHER(condition="Sunny",temperature="32ºC",trend="high");
VESSELS IN PORT(quantity="350",trend="low",days max="30")
```

Based on the selected content, we grouped the sets with the same combination of intent messages. In total, 15 distinct sets of intent messages were selected for each domain, and the author verbalized each of them in Brazilian Portuguese. All the verbalizations were made based on a small sample of 100 texts per domain. The syntactic and lexical patterns in the texts present in the chosen samples were used to produce a variety of target intent verbalizations, and intermediate representations in the pipeline steps were manually annotated. After implementing the process to daily collect, validate, and store data on a structured format, we compared and evaluated texts generated by the three most commonly used NLG automated journalism approaches: template, pipeline and neural end-to-end.

3.2 Template Architecture

Template-based data-to-text NLG systems directly translate non-linguistic input to linguistic structure by filling gaps in predefined template texts (REITER, 1995), and often have only two modules: content selection and textual realization. Because only the predefined variables can change in static templates, problems with maintainability and scalability arise from this approach; static template-based systems cannot be readily used to address discourse ordering, sentence formation and aggregation, referring-expression generation and lexicalization. This can result in text that lacks creativity or personalization, as it relies heavily on pre-written templates. Additionally, it may not be suitable for generating complex or nuanced text, as the templates may not be able to account for all possible variations or scenarios.

The main advantage of template-based approaches over other more sophisticated NLG architectures is seen in cases where good linguistic rules are not yet available or in conditions where consistency and structure are key. Some examples of template-based systems that publish daily reports on Twitter in Portuguese are @RosieDaSerenata, which identifies expenses with discrepancies and indicates the reasons that lead it to believe they

are suspicious, and @ruibarbot, which monitors stalled processes in the Supreme Federal Court of Brazil (STF) (FURTADO, 2020). Some examples in English are Editing TheGrayLady, which highlights changes to news on main page of The New York Times newspaper, and @snippet_jpg, which posts snippets from newspaper front pages published more than 100 years ago.

3.3 Pipeline Architecture

The pipeline architecture for NLG involves a series of stages or components that work together to transform input data into natural language output. The most frequently used NLG pipeline approach converts structured input data to output text in six steps: *Content Selection, Discourse Ordering, Text Structuring, Lexicalization, Referring Expression Generation and Textual Realization* (HORACEK, 2001).

Content Selection In the first step of a pipeline architecture for natural lanugage generation, the Content Selection module decides which information should be communicated in the text. It is a critical part of the pipeline approach, because generating texts that are relevant and informative is critical for effective communication. In NLG systems, content selection can be performed using various techniques, such as rule-based approaches, statistical models, or machine learning algorithms. These techniques can take into account various factors such as the intended audience, the purpose of the text, the context of the communication, and the available data sources.

Content selection is typically done by analyzing the input data, which can include structured data (e.g., databases, spreadsheets) or unstructured data (e.g., text, audio, images), and extracting the most relevant information. Content selection can be performed using various techniques, such as rule-based approaches, statistical models, or machine learning algorithms. These techniques can take into account various factors such as the intended audience, the purpose of the text, the context of the communication, and the available data sources.

For instance, in a weather report generation system, content selection might involve analyzing the current weather conditions, the forecasted weather patterns, and any relevant historical data to determine what information should be included in the generated report. The system might also take into account the location of the intended audience, the time of day, and other factors to generate a report that is tailored to the needs of the user.

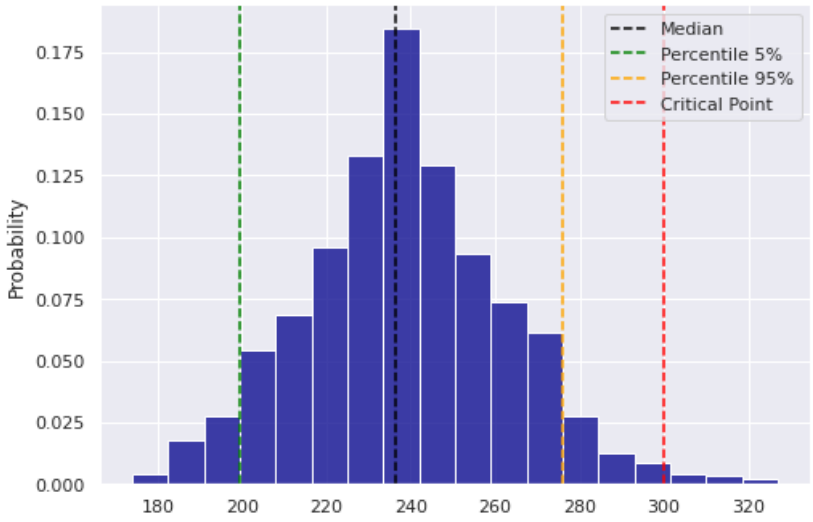

Figure 16: Histogram of the average number of vessels at the port of Santos during a period of 6 months. Data obtained from the Marine Traffic website.

Since content determination precedes language generation, template-based systems can treat it similarly to the pipeline-base systems. While the former tend to take their departure from structured database records, the latter often use richer input, where some decisions concerning linguistics have already been made (DEEMTER; THEUNE; KRAHMER, 2005). This step often requires the assistance of a domain expert to understand which information is relevant in the context and how to group them into intent messages. This means that the message-generation process and the form and content of the generated messages are highly application-dependent.

Intents in the context of the Blue Amazon are, for example, when a new earthquake is detected, when the total number of vessels in a given port hits a new record or when temperature hits its lowest point in the winter. Figure 16 shows the histogram of the average number of vessels at the port of Santos, registered during a period of 6 months. If the number of vessels is in the bottom 5% percentile, or in the top 5% percentile, then the corresponding intent is raised. Additionally, if the number of vessels reaches a critical level of more than 300 (determined by our domain expert), then another intent is raised and used as input by the next module.

To create the intents in our Blue Amazon automated journalism system, we studied the relevant time-series data using insights from oceanography researchers. The intents are triggers that may apply depending on the data being analyzed; for example, when oil extraction reaches a critical level.

The following text is an example of the content selection module output for our system:

```
LOCATION(city="Santos",uf="SP",timestamp="Jan 15, 2022");
WEATHER(condition="sunny",temperature="32ºC");
VESSELS IN PORT (quantity="350",trend="high",days max="30");
OCEAN(fishing condition="excellent",height of the sea:"1.8 meters");
```

Discourse Ordering Once the relevant content is selected, the next step of the pipeline is Discourse Ordering, which is responsible for imposing order and structure over the set of messages to be conveyed (HEILBRON et al., 2019). The information is presented in a specific order, and there is typically an underlying tree structure to the presentation, so a text is not merely a random collection of facts. In the simplest possible terms, this is comparable to a story with a beginning, middle, and end; however, most papers have much more structure than this. Reading a version of a newspaper story where the sentences and paragraphs have been randomly rearranged is much more complicated than reading a well-structured text.

Although some authors have had success with machine learning solutions to order facts for discourse planning (DIMITROMANOLAKI; ANDROUTSOPOULOS, 2003), most applications utilize a rule-based approach. For example, a possible outcome order might be: `LOCATION, TEMPERATURE, EXCELLENT WEATHER AND FISHING CONDITIONS → CAUSES → PEAK OF VESSELS IN PORT, OIL EXTRACTION`

Text Structuring Also referred to as Sentence Aggregation by some authors, Text Structuring is a NLG sub-task in which intents are organized into sentences and paragraphs. Given a linearized set of intent messages, the goal of this step is to generate predicates segmented by sentences. While it is possible to use a dedicated attention mechanism (JURASKA; WALKER, 2021), most NLG systems utilize explicit content text structuring.

For the case of our automated journalism agent, a possible text structure for the output of this module might be:

```
Paragraph 1:  LOCATION, TEMPERATURE
Paragraph 2:  EXCELLENT WEATHER AND FISHING CONDITIONS → CAUSES → PEAK OF VESSELS IN PORT
Paragraph 3:  OIL EXTRACTION
```

Lexicalization The process of selecting the appropriate words and phrases from a vocabulary or lexicon to convey a message is known as Lexicalization. It involves mapping concepts or ideas from the underlying data or input to their corresponding linguistic expressions, ensuring that the generated text is accurate and fluent. Lexicalization is performed by adding words, phrases or word patterns to a language's vocabulary in order

to inflect words based on their grammatical use (tense, number, case and gender, for example) (STEDE, 1994). For example, in the sentence "The cat chased the mouse", the process of lexicalization would involve selecting the words "cat", "chased", and "mouse" from a lexicon and combining them into a grammatically correct sentence. The choice of words may depend on various factors, such as the context, the target audience, and the tone or style of the text.

In many cases, this step can be done trivially by hard-coding specific words or phrases for each domain concept. In some cases, however, fluency can be improved by allowing the NLG system to vary the words used to express a concept or relation, either to achieve variety or to accommodate subtle pragmatic distinctions (DAUMÉ III et al., 2002). For our Blue Amazon agent, the system chooses a lexicalization template for each structured sentence. These templates are chosen from a list of sentence templates, created by the author, assuring that there are multiple options to choose from. The templates provide for gender and number inflection, e.g., "Rio de Janeiro accumulates 1 day with vessel overflow in the port" vs. "Rio de Janeiro accumulates 5 day with vessel overflow in the port". A fill-template would not take into account number and gender inflection, decreasing variety in the output text.

Referring Expression Generation In order to replace entity tags in templates, this module is responsible for generating noun phrases to refer to entities mentioned as discourse unfolds. There are neural-based approaches that generate referring expressions for entities not found during the training process, but we used a list of possible expressions for each entity. For the first reference to an entity in the text, a full description is used (e.g., `WEBSITE → "Marine Traffic"`), whereas for subsequent references a random referring expression to the entity is chosen (e.g., `WEBSITE → "the website; "the site"; "the Marine Traffic website"; "it"`; etc.). This process generally takes into account the contextual factors involved, including in particular the content of previous generated sentences or texts. In our work, we enumerated the most common entities in the Blue Amazon domain and wrote down their possible referring expressions, which are picked at random.

Textual Realization The last step of the pipeline approach performs the remaining adjustments to transform intermediate machine representations into text. This is performed by applying grammar rules in order to produce a text that is syntactically, morphologically, and orthographically correct. Detokenization, contractions, nominal and verbal are performed in order to make the content grammatically consistent. For our robot-journalism agent, this step applies a final layer of textual manipulation, to make the content more appealing to the target audience. This is done by adding customized

greeting messages and emojis. The resulting texts are published every day using Twitter's API.

Overall, the pipeline architecture for NLG is designed to take structured data and transform it into natural language output that is both informative and easy to understand for the target audience. The main advantages of this approach are: more fluency and lexical variety than the template approach, interpretability and no risk of data hallucination, which can be problematic in sensitive domains. However, this approach usually demands extensive manual annotation and text variety depends on the diversity of the annotated corpus. An example of pipeline-based automated journalism system which publishes daily reports on Twitter in Portuguese is @DaMataReporter, a robot-journalist system covering the Brazilian Amazon deforestation (TEIXEIRA et al., 2020). An example in English is @earthquakeBot, which publishes tweets about any detected earthquakes with magnitude of 5.0 or greater in real time.

For our Blue Amazon reporter, the grammar used by the pipeline approach was built by running the content selection in the collected dataset, and manually verbalizing the non-linguistic reports. The input and output representations for each pipeline module were manually annotated by the author, and this process resulted in a list of possible discourse orders, text structures, lexicalizations, and referring expressions to be used when generating output text.

3.4 Neural End-to-End Architecture

Neural end-to-end architectures for natural language generation recently gained popularity due to massive amounts of data and computational power now available (CHEN; LIN, 2014). Given enough labeled data, it does become possible to learn a mapping function which converts non-linguistic input into human-readable text without explicit use of intermediate representations. Such architectures operate by applying deep neural networks, convolutional neural networks, recurrent neural networks (RNN) and transformers (LI, 2017). Successful examples of end-to-end robot-journalism applications can be found in contexts where making mistakes and hallucinating content is not critical, for example data storytelling (AMMANABROLU et al., 2019) and image captions (HE; DENG, 2018).

A neural approach to NLG involves using deep learning algorithms to generate natural language output from structured data or information. This approach involves the

following steps:

1. Data preparation: The first step in a neural NLG approach is to prepare the input data. This could involve encoding the input data in a format that can be processed by a neural network, such as a sequence of vectors or a matrix.

2. Neural network architecture: The next step is to design the architecture of the neural network. This involves selecting the type and number of layers in the network, the activation functions to be used, and other hyperparameters.

3. Training: The neural network is then trained on a dataset of input-output pairs. The network learns to map the input data to the corresponding output text by adjusting the weights of the connections between its neurons.

4. Evaluation: Once the network has been trained, it is evaluated on a held-out dataset to assess its performance. Various metrics can be used to evaluate the network, such as accuracy, precision, recall, or specialized NLG metrics, such as BLEU, ROUGE, and METEOR.

5. Deployment: The trained neural network model can be deployed to generate natural language output for new input data. Machine learning models are usually deployed in a specialized cloud environment and accessed via API.

6. Monitoring: Finally, the must be monitored in the production environment, to make sure that it is not generating false information and its resulting metrics are close to the training metrics. If that is not the case, the model must be retrained with new data.

Our goal here was to learn a direct mapping from a intent-attribute-value input text in Portuguese to a human-readable output text also in Portuguese. We tested four text-to-text transformer-based models: Bart, T5, Blenderbot, and GPT2. While GPT2 utilizes a decoder only module, the first three models use a encoder-decoder scheme (CHO et al., 2014). Although the encoder-decoder architecture was initially developed for machine translation, it has proven successful at related sequence-to-sequence prediction problems such as text summarization, question answering and computer vision.

Encoder-decoder networks have two distinct modules: the encoder, which transforms the input sequence into one feature vector, and a decoder which generates the output sequence. Figure 17 shows an example of this process in the context of our automated journalism agent. The system receives the following intent-attribute-value input text:

Location(city="Rio de Janeiro", state="RJ");

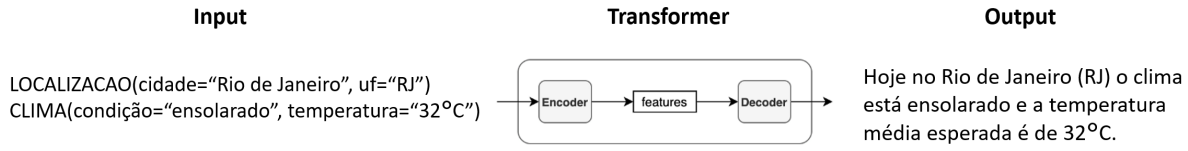

Figure 17: Example of a generated text for our automated journalism agent using transformers.

Weather(climate="sunny", temperature="32º C")

and outputs the following text (here translated to English):

Today in Rio de Janeiro (RJ), the weather is sunny, and the average temperature expected during the day is 32º C.

For all neural end-to-end models used in this work, the inputs are the same intent-attribute-value textual representations written in Brazilian Portuguese. To train the machine learning models, we used the resulting text of the pipeline architecture as ground truth.

The main advantages of the neural end-to-end approach are: the ability to generalize to unseen contexts and different inputs, and quick implementation given enough labeled data is available for training machine learning models. However, such methods have a high risk of data hallucination, require extensive manual annotation, and are often difficult to audit due to subjective evaluation metrics. The most notorious example of system which employs neural end-to-end concepts is Arria's NLG Engine, a cloud-based enterprise software platform that automatically recognizes patterns in large volumes of complex data (SRIPADA et al., 2014). Arria's system is able to derive patterns, facts and insights before structuring this information in the best possible manner, and it is already being used to generate narrative reports to optimise agricultural yield potential, as well as providing data intelligence to aircraft engine maintenance staff.

3.5 Experiments

Both our automated journalism agent and the experiments in this work have been implemented using the Python programming language. We compared the template, pipeline

and neural end-to-end approaches on our Blue Amazon dataset, in order to understand when it would be better to use each of them. For both the template and pipeline approaches, the input was a tabular data containing all the Blue Amazon information collected, as shown in Table 1. As for the neural end-to-end approach, the input was the intent-attribute-value semantic representation of the tabular data, while the target of the models was the resulting text from the pipeline approach. For our experiments, we used the following 4 transformer models: BERT, T5, Blenderbot, and GPT2. Data from 3 different domains (weather, tides chart and marine traffic) was randomly split into training (60%), validation (20%) and testing (20%). The whole dataset contained a total of 2000 pairs of input data and target text, where ground truth used to train the machine learning models was the output from the pipeline approach. We also performed an additional experiment using an extra domain (reported earthquakes on the Brazilian coast) with the goal to measure how well each architecture would perform for a scenario not seen on the training dataset (out-of-domain validation).

All the experiments using neural networks were performed using a Google Colab with a NVIDIA Tesla K80 GPU for a maximum of 100 epochs, with a learning rate of 1e-5, a batch size of 10 samples and an early stopping criteria of 3 epochs. Results were also evaluated by collecting feedback from 10 masters and phd researchers from the oceanography course at the University of São Paulo (USP), aged between 22 and 35 years old, during March 2023.

For each pair of input data and output text, the participants rated the output based on the fluency (i.e., *"is the text easy to read?"*), semantics (i.e., *"does the text clearly express the input data?"*) and lexical variety (i.e., *"is the text original or is the content being repetitive?"*), using a 1-5 Likert scale (JOSHI et al., 2015), where 1 means strongly disagree and 5 means strongly agree. The neural end-to-end models were also evaluated using seven different NLG automatic metrics: BLEU, ROUGE, METEOR, GLEU, BLEURT, COMET, and PRISM.

For the neural end-to-end approach, we performed one additional experiment: we replaced all the numbers with their textual representation (eg. 25 becomes twenty five), to validate if this would lead to better overall results by reducing the probability of data hallucination (NIE et al., 2019). We also published 1.000 tweets using each architecture, in order to validate which of them would result in more likes and positive user interaction, and whether the result would correlate with the evaluation metrics.

Finally, in order to further evaluate the results obtained by each architecture, we

followed the methodology proposed by Thomson et al. (THOMSON; REITER, 2020) in 2020. Their goal was to develop techniques which could be used to evaluate accuracy in longer texts which communicate complex data and possibly insights derived from the source data. Their approach was to ask multiple human annotators to identify specific errors in a text, and categorise the errors into one of a small number of types. The errors are divided into one of the following six categories:

- Incorrect number: This checks for wrong data in the output texts and includes both digits and numbers which are spelled out.
- Incorrect named entity: This includes places, names of people, companies, and days of the week.
- Incorrect word: Any word which is not one of the above items and it is incorrect.
- Context error: A phrase which causes an incorrect inference because of context or discourse.
- Not checkable: A statement which can not be checked; either the information is not available or it is too time-consuming to verify.
- Other: Any other type of mistake that may happen in the output text. The authors claim that this category should only be used as a last resort.

4 RESULTS

4.1 Human Evaluation Metrics

Table 2 depicts the results of the human evaluation metrics for the template, pipeline, and different neural end-to-end architectures which were studied in this work. The evaluations were carried on by 10 masters and phd researchers from the oceanography course at the University of São Paulo (USP). Each of the, looked at a sample of 100 random texts generated by our system and evaluated them using a 1-5 Likert scale, where 1 means strongly disagree and 5 means strongly agree.

Results show that the template-based architecture outperformed the others architectures in both fluency and semantics, with scores of 4.7 and 4.6, respectively. However, it received a very low score of 3.3 for lexical variety, which happens due to the nature of the architecture, which often sacrifice lexical variety for fluency and semantic (PEREIRA; TEIXEIRA; PINTO, 2015). This architecture is usually difficult to maintain due to the fact that it requires extensive domain knowledge and dedicated human annotators. Texts generated by this approach also don't scale very well, because new data and new domains require more rules to be created. The result of lacking lexical variety is that the texts generated by this architecture are usually very repetitive, as shown in Table 4. This is a known limitation of the approach, hence we decided to use it only for sensitive scenarios, where communicating the message on a very fluent and objective way is more important than having well connected sentences and lexical variety. Examples of these scenarios in the Blue Amazon domain are, for example, when a new earthquake is detected or when oil extraction reaches a critical level.

Although the pipeline-based architecture presented less fluency and semantics than the template-based approach, it also obtained high scores in those metrics, and it was evaluated with the best overall score for lexical variety. This means that in domains where there are enough linguistic insights and computational resources available to develop a pipeline-based architecture, and also there is no critical or sensitive information to be

Table 2: Results for the human evaluation metrics for each architecture, using the Blue Amazon as a common domain.

Architecture	**Fluency**	**Semantics**	**Lexical Variety**
Template	**4.7**	**4.6**	3.3
Pipeline	4.5	4.3	**4.4**
End-to-end (Bart)	4.3	4.1	3.9
End-to-end (T5)	4.0	3.9	3.7
End-to-end (Blenderbot)	3.2	3.1	3.1
End-to-end (GPT2)	2.3	2.2	2.9

conveyed, the pipeline-based architecture is a better candidate than the template-based approach, because it provides for less repetitive texts to the target audience with good results for both fluency and semantics. It also has the advantage of not hallucinating data and not needing to train and evaluate a machine learning model, unlike the end-to-end approaches.

As for the neural end-to-end architecture, results show that all the four tested transformer-based models scored less in all the human quantitative metrics for the case where there is no unseen scenario present in the test data. However, it is interesting to note that both the Bart and the T5 neural end-to-end architectures presented results close to the Pipeline approach. Unlike the other three transformer models, GPT2 utilizes a decoder only module; as such, it failed to complete long sentences and properly transform input data into coherent output text, which justifies the low scores it received.

Deep learning methods for NLG often hallucinates data and do not convey all the meaningful information in the input, as shown in Table 4. Such approaches usually make more lexical and semantic mistakes compared to the pipeline-based architecture, given that the latter has a dedicated lexical module while the former does not. The main advantage in this approach is in domains where hallucinating data is not critical or where there are no domain experts to provide linguistic insights, which is not the case for our automated journalism agent. Such systems also thrive when there are new scenarios present in the data, because it generalizes better than the template and the pipeline approach due to the nature of such systems, as shown in Table 6.

Table 3: Results for the automatic evaluation metrics for each architecture, using the Blue Amazon as a common domain.

Architecture	BLEU	ROUGE	METEOR	GLEU	BLEURT	COMET	PRISM
End-to-end (Bart)	**50.2**	78.1	71.8	**49.9**	**63.5**	**59.5**	**56.5**
End-to-end (T5)	47.3	**79.2**	**72.3**	46.8	63.1	55.7	53.5
End-to-end (Blenderbot)	39.2	42.1	58.9	39.6	51.3	46.3	42.9
End-to-end (GPT2)	19.4	17.4	19.8	18.8	19.4	17.8	17.2

4.2 Automatic Evaluation Metrics

Table 3 depicts the results of the automatic evaluation metrics for the four neural end-to-end architectures which were studied in this work. Evaluation was performed on the test dataset (20.000 texts) using seven different NLG metrics: BLEU, ROUGE, METEOR, GLEU, BLEURT, COMET, and PRISM. The results for the template and pipeline metrics are not shown here, because they were used as ground truth for training the neural end-to-end models.

Results show that the Bart neural end-to-end model outperformed the other neural end-to-end models for the BLEU, GLEU, BLEURT, COMET, and PRISM automatic metrics, while the T5 model achieved the best ROUGE and METEOR scores. While Blenderbot obtained average scores, GPT2 obtained the lowest scores overall for all the purposed metrics because it utilizes a decoder only module. Works in simmilar contexts have reported BLEU scores ranging from 45% to 65%, ROUGE scores ranging from 65% to 85% (DUŠEK; NOVIKOVA; RIESER, 2018), and both PRISM and COMET scores ranging from 30% to 60% (PU et al., 2021).

Table 4 shows some examples of input and output pairs for each of the compared architectures. The template-based approach outputs texts on a very objective and straightforward way, while the pipeline-based approach outputs more complex text with connected sentences and lexical variety. The neural-based approach sometimes hallucinates data, and also makes both lexical and semantic mistakes, as shown in Table 4.

Another interesting conclusion in our work was the fact that the data augmentation process helped to improve the neural end-to-end model performances. By using data augmentation, we obtained more data to train the neural-based machine learning models

For example, the Bart model showed an increase in the BLEU score from 49.6 to 50.2, and the ROUGE score increased from 77.6 to 78.1, which is a slight improvement.

4.3 Types of Errors

Following the methodology proposed by Thomson et al. (THOMSON; REITER, 2020) in 2020, we looked into a sample of 100 texts generated by each of the studied architectures, and manually annotated errors sorting them into one of the following five categories: incorrect number, incorrect named entity, incorrect word, context error, and not checkable. The results are shown in Table 5.

The total amount of errors that we found on the texts generated by each architecture correlates with both the human evaluation metrics and the automatic evaluation metrics. The template architecture presented less errors than all the others, with a total of 6 errors in a sample of 100 texts. The pipeline architecture presented a total of 10 errors, while the neural end-to-end Bart model presented 15. We also found 18 errors for the T5 model, 27 for the Blenderbot model and, finally, GPT-2 presented the highest amount of mistakes, with a total of 31 errors found.

The template architecture presented less errors because the system was developed using simple and straightforward rules, created by the author with the aid of domain specialists, and its mistakes were due to incorrect communications between the content selection and the textual realization modules. Out of the 6 errors, 4 were due to wrong use of words, and 2 to wrong context, and it is interesting to note that there were no numeric neither name errors due to the nature of the architecture. Word errors happened due to lexicalization mistakes in the rules, for example a text said that "there are currently 1 cargo ships in the Antonina Port (SC)", mismatching singular and plural forms.

The pipeline architecture made a total of 10 errors, being 2 word errors, 4 name errors, 3 context errors, and 1 not checkable error. Just like the template approach, the pipeline architecture also made no numeric mistakes. Two texts generated by the pipeline approach had lexicalization issues, and it also got confused with named entities. One of the context errors happened when the system wrote that it was a peak temperature for the city during that year, but in fact it was only the peak temperature for the current month. Finally, the not checkable error happened when it said that the Santos port had its highest ever amount of vessels in the port, but it couldn't claim that when it didn't have the whole historical data availabe, only a few months.

The Bart model performed best out of the four neural end-to-end architectures which were trained using machine learning and transformers, and this result correlates with the evaluation metrics. Bart made a total of 17 errors, being 6 numeric errors, 3 word errors, 3 name errors, 2 context errors, and 3 not checkable errors.

The T5 neural architecture made a total of 20 errors, being 8 numerical errors, 4 word errors, 2 name errors, 4 context errors, and 2 not checkable errors. The Blenderbot architecture made a total of 28 errors, being 11 numercial errors, 5 word errors, 4 name errors, 4 context errors, and 4 not checkable errors. Finally, the GPT2 architecture made a total of 32 errors, being 13 numercial errors, 5 word errors, 5 name errors, 4 context errors, and 5 not checkable errors. It is interesting to note that the neural end-to-end architectures hallucinated numbers and words multiple times, which is a known issue for this kind of NLG automated journalism approach.

4.4 Out of Domain Evaluation

Table 6 depicts the results for fluency and semantics, as well as the three most frequently used NLG automatic evaluation metrics (BLEU, ROUGE, and METEOR) for the template, pipeline and two neural end-to-end architectures, Bart and T5 (which presented better overall results), for a out-of-domain test dataset containing only information about detected earthquakes on the Brazilian coast. The goal of this experiment was to measure how well each system would generalize to an unseen scenario, with different content and data.

The results presented in Table 6 show that, for an unseen domain, the results for the template, pipeline and neural end-to-end architectures become very similar. The Bart model presented results better than the others for fluency, semantics, and BLEU, while the T5 presented better results for the ROUGE and METEOR metrics. This results is very important for our work, because it shows that the neural end-to-end methods perform slightly better when used on an unseen domain where there are no rule-based guidelines to generate text. This means that, for our Blue Amazon domain, it is advantageous to use the neural end-to-end models in practice when data from a new domain is fed to the content analyzer.

4.5 Converting Numbers to Text

Motivated by results in other NLP researches (THAWANI et al., 2021), we also studied the effect of replacing all the numbers with their textual representations (eg. 25 becomes twenty five) to validate if this would reduce the occurrence of data hallucination for the neural end-to-end models. We compared the three main automatic NLG metrics (BLEU, ROUGE, and METEOR) for the Bart neural end-to-end architecture, because it performed best in our experiments.

The results presented in Table 7 show replacing all the numbers with their textual representation does increase the results from the three main NLG metrics. When inspecting the generated texts, it was clear that the reason for this was the lower occurrence of data hallucination.

4.6 Twitter Engagement

To conclude our experiments, Table 8 depicts the engagement results for the texts published by our Blue Amazon agent. We manually analyzed user interactions with the tweets for a period of 8 consecutive weeks. We published 1.000 tweets using each of the template, pipeline and neural end-to-end approaches, in order to validate which architecture would result in more likes and positive user interaction. For the neural end-to-end approach, we opted to use the Bart model, because it showed better overall results for our Blue Amazon domain.

The results presented in Table 8 show that the pipeline architecture received more likes than the other two approaches, with a total of 105 likes, while the template method received 82 likes and the neural end-to-end method received 95 likes. What is interesting to note is that, although the tweets generated by the Bart model received fewer likes than the pipeline approach, Twitter users interacted more with it and commented 31 times. This happened because sometimes it generated inconsistent data (for example, a temperature of 75°C, and incorrectly saying that Recife is located in Rio de Janeiro), and thus some tweets received negative comments from the audience. Overall, the engagement result correlate with the evaluation metrics.

Table 4: Examples of input and output pairs for some of the compared architectures for automated journalism.

Architecture	**Input**	**Output**	**Conclusion**
Template	`LOCALIZAÇÃO(cidade= "Santos", uf="SP"); CLIMA(condição="nublado", temperatura="26ºC", vento="18km/h"); TERREMOTO(magnitude="1.3 mR", depth="10km")`	*Hoje em Santos (SP) a previsão é de tempo nublado. A temperatura é de 26ºC. O vento é de 18km/h. Foi detectado um terremoto de magnitude 1.3 mR e profundidade de 10km.*	The architecture managed to accurately convey all the information present in the input data without any lexical or semantic mistake. However, the text is too simple, lacks connection between sentences and does not provide for lexical variety.
Pipeline	`LOCALIZAÇÃO(cidade= "Cabo Frio", uf="RJ"); CLIMA(condição= "ensolarado", temperatura="33ºC", mar="0,8m", umidade="76%", nebulosidade="54%", vento="29km/h", protetor="sim")`	*Hoje a previsão é de tempo ensolarado, com uma temperatura média esperada de 34ºC. A umidade de 54%, a nebulosidade de 29km/h e a velocidade do vento de 54%. Utilize protetor solar se for sair de casa!*	The architecture managed to accurately convey all the information present in the input data without any lexical or semantic mistake, and also provided for well connected sentences.
End-to-end (Bart)	`LOCALIZAÇÃO(cidade= "Itajaí", uf="PE"); CLIMA(condição= "ensolarado", temperatura="25ºC"); NAVIOS(quantidade="185", dias max="28")`	*Hoje em Itajaí (PE) foi detectada uma quantidade de 285 navios no porto da cidade, essa e esse é o maior valor registrado nos últimos 28 dias.*	The architecture output a text with correct semantic and fluency, but forgot to convey weather related data and made a mistake in the amount of vessels in the port.
End-to-end (T5)	`LOCALIZAÇÃO(cidade= "Itajaí", uf="PE"); NAVIOS(quantidade="180", temperatura="35ºC", mar="0,8m", umidade="76%", vento="29km/h", protetor solar="sim")`	*Hoje a previsão é de tempo temperatura média esperada de 35ºC a umidade é de 29km/h a velocidade do vento é de 76%. Utilize protetor solar!*	The architecture output a text with some lexical and fluency mistakes, failing to properly connected sentences. It also swapped the values of humidity and wind speed, which would be critical on a more sensitive scenario.
End-to-end (GPT2)	`LOCALIZAÇÃO(cidade= "Recife", uf="PE"); CLIMA(condição= "ensolarado", temperatura="25ºC", mar="1,8m", umidade="58%, vento="21km/h")`	*Hoje aaa temperatura é de 25°V, aaa temperatura é de 25°V, aaa temperatura é de 25°V.*	The architeture swapped °C with °V for no particular reason and failed to convey most of the information in the input data. It also repeated the temperature value three times and output nonexistent words.

Table 5: Error evaluation for each architecture, using the Blue Amazon as a common domain.

Architecture	Total	Number	Name	Word	Context	Not Checkable
Template	6	0	0	4	2	0
Pipeline	10	0	2	4	3	1
End-to-end (Bart)	17	6	3	3	2	3
End-to-end (T5)	20	8	4	2	4	2
End-to-end (Blenderbot)	28	11	5	4	4	4
End-to-end (GPT2)	32	13	5	5	4	5

Table 6: Results for both fluency, semantics and the three most frequently used NLG automatic evaluation metrics for each automated journalism architecture, using an out-of-domain dataset.

Architecture	Fluency	Semantics	BLEU	ROUGE	METEOR
Template	4.1	4.1	47.8	76.1	70.5
Pipeline	4.0	4.1	47.3	76.9	70.8
End-to-end (Bart)	**4.3**	**4.2**	**49.3**	77.2	71.3
End-to-end (T5)	4.0	3.9	47.3	**78.2**	**71.9**

Table 7: Comparison between the results for the three most frequently used NLG automatic evaluation metrics using the Bart neural end-to-end method, with and without replacing numbers with their equivalent textual representation.

Architecture	BLEU	ROUGE	METEOR
End-to-end (Bart) Without Numeric Replacement	50.2	78.1	71.8
End-to-end (Bart) With Numeric Replacement	50.4	78.5	72.1

Table 8: Twitter engagement results for the template, pipeline, and neural end-to-end (Bart) automated journalism architectures for 1.000 tweets published by each method during a period of 8 consecutive weeks.

Architecture	Likes	Total Comments	Positive Comments	Neutral Comments	Negative Comments
Template	82	20	5	14	1
Pipeline	105	31	13	16	2
End-to-end (Bart)	95	37	10	14	13

5 CONCLUSION

5.1 Discussion

We have developed a publicly available Blue Amazon dataset and an automated journalism agent which publishes daily Brazilian Portuguese reports by collecting, storing and analyzing information from multiple publicly available sources containing data related to the Brazilian coast. Our Blue Amazon agent, named BLAB (BLue Amazon Brain) Reporter, combines different natural language generation techniques for automated journalism to transform structured input data into comprehensive, fluent, and semantically correct Brazilian Portuguese news and reports which are published on Twitter every day.

Our work also compared the three most frequently used automated journalism architectures: template, pipeline, and neural end-to-end, using the Blue Amazon as a common domain. Human evaluation metrics showed that the template-based architecture performed best in both fluency and semantics, but the texts generated by this approach lacked lexical variety and were labeled as repetitive by domain experts. The pipeline architecture also obtained high scores for fluency and semantics, while providing for more lexical variety than the template approach. The target audience also interacted more with tweets generated by the pipeline approach, and the majority of the comments on Twitter were considered positive. Although the Bart and T5 neural end-to-end models presented results close to the template and the pipeline architectures, they scored less in all the evaluation metrics. Deep learning methods for NLG often hallucinate context and data, and also do not convey all the meaningful information in the input. They also make more lexical and semantic mistakes than the pipeline-based architecture, given that the latter has a dedicated lexical module while the former does not.

However, when testing on an unseen domain, both the Bart and T5 neural end-to-end architectures generalized better than the template and the pipeline architectures, presenting higher overall scores. While the template approach does not have specific rules for the new scenarios and the pipeline approach does not know how to select content,

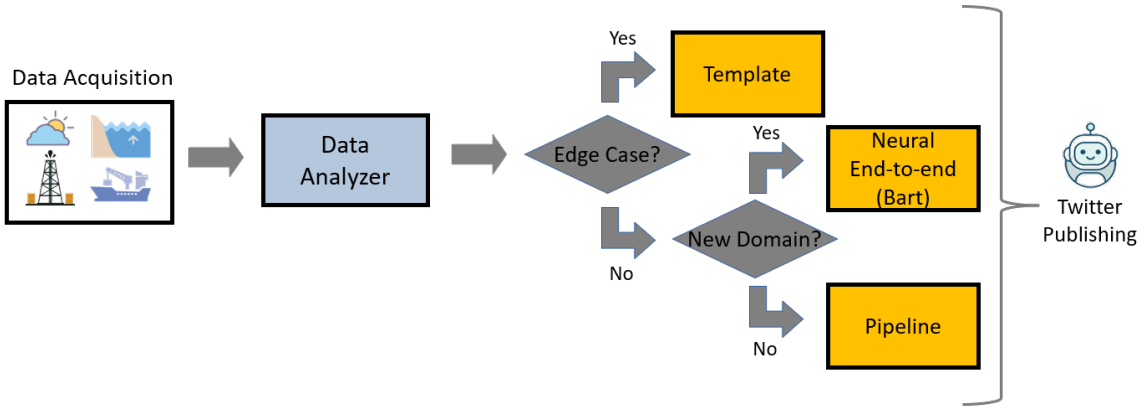

Figure 18: System architecture for our Blue Amazon agent, developed to extract information from multiple publicly available sources, store and analyze data, and publish daily Brazilian Portuguese reports about the Blue Amazon on Twitter.

lexicalize or structure the resulting text, the neural approaches were able to learn from annotated data for other domains, and generalize better to an unseen domain by conveying better texts. We also concluded that, by replacing all the numbers with their textual representation (e.g. 25 becomes twenty five), we obtained better overall results by reducing the probability of data hallucination.

Figure 18 shows the architecture of the automated journalism agent we adopted after our experiments. When the data is collected and stored on our system's database, a rule-based data analyzer module decides whether the content is critical or not. If that is the case, our Blue Amazon agent employs the template approach in order to communicate the message in a straightforward manner, avoiding the risk of publishing texts with poor fluency or semantics. If the data analyzer module decides that the information is not critical, our system checks whether the structured data contains information from an unseen domain. If that is the case, the neural end-to-end Bart model is used to generate the resulting text; otherwise our system uses the pipeline approach.

It is important to emphasize that generating meaningful, semantically correct, and accurate text from structured input data is still a very complex task, requiring specific domain knowledge, human annotations, data storage, feedback from specialists, and experimenting with different architectures. This is a process that goes beyond simply applying natural language generation algorithms. By providing all the annotated data and codes which were used for our project, as well as the natural language generation architectures developed for the Blue Amazon domain, we hope to help future NLG endeavors and also encourage more people to develop Brazilian Portuguese automated journalism applications.

5.2 Future Work

Some ideas were not explored in this work, but we believe they should be explored in future work:

- Adding more sources of information to the Blue Amazon agent, such as real time reporting of illegal fishing activities, navy information, and other natural disasters. We believe this will improve the quality and the diversity of the texts generated by our automated journalism agent.
- Using transfer learning to generate a richer artificial dataset. We obtained slightly better overall results by using data augmentation techniques, and it would be interesting to validate if transfer learning would be able to increment the evaluation metrics.
- Testing and incorporating ChatGPT or similar large language models to our automated journalism agent, to test if it would yield better overall texts for our Blue Amazon domain.
- Trying different neural end-to-end machine learning models at each step of the pipeline architecture, to validate whether this would lead to better results.
- Experimenting with text summarization architectures to outline public news related to the Blue Amazon. When talking with our domain specialists, they suggested that summarizing news related to the Blue Amazon would be interesting to increase user engagement on Twitter.

REFERENCES

AMMANABROLU, P.; TIEN, E.; CHEUNG, W.; LUO, Z.; MA, W.; MARTIN, L.; RIEDL, M. Guided neural language generation for automated storytelling. In: *Proceedings of the Second Workshop on Storytelling*. [S.l.: s.n.], 2019. p. 46–55.

ARUN, A.; BATRA, S.; BHARDWAJ, V.; CHALLA, A.; DONMEZ, P.; HEIDARI, P.; INAN, H.; JAIN, S.; KUMAR, A.; MEI, S. et al. Best practices for data-efficient modeling in NLG: how to train production-ready neural models with less data. *arXiv preprint arXiv:2011.03877*, 2020.

AUER, S.; BIZER, C.; KOBILAROV, G.; LEHMANN, J.; CYGANIAK, R.; IVES, Z. Dbpedia: A nucleus for a web of open data. In: *The Semantic Web*. [S.l.]: Springer, 2007. p. 722–735.

BALDWIN, J.; CHANNARUKUL, S. Domain-oriented two-stage aggregation: generating baseball play-by-play narratives. In: IEEE. *2015 7th International Conference on Knowledge and Smart Technology (KST)*. [S.l.], 2015. p. 42–47.

BELZ, A.; REITER, E. Comparing automatic and human evaluation of NLG systems. In: *11th Conference of the European Chapter of the Association for Computational Linguistics*. [S.l.: s.n.], 2006. p. 313–320.

BHANDARI, R. S.; BANSAL, A. Impact of search engine optimization as a marketing tool. *Jindal Journal of Business Research*, SAGE Publications Sage India: New Delhi, India, v. 7, n. 1, p. 23–36, 2018.

CAMPOS, J.; TEIXEIRA, A.; FERREIRA, T.; COZMAN, F.; PAGANO, A. Towards Fully Automated News Reporting in Brazilian Portuguese. In: SBC. *Anais do XVII Encontro Nacional de Inteligência Artificial e Computacional*. [S.l.], 2020. p. 543–554.

CASACUBERTA, F.; CIVERA, J.; CUBEL, E.; LAGARDA, A. L.; LAPALME, G.; MACKLOVITCH, E.; VIDAL, E. Human interaction for high-quality machine translation. *Communications of the ACM*, ACM New York, NY, USA, v. 52, n. 10, p. 135–138, 2009.

CASTELVECCHI, D. Are ChatGPT and AlphaCode going to replace programmers? *Nature*, 2022.

CASTRO, B. M.; BRANDINI, F. P.; DOTTORI, M.; FORTES, J. F. A Amazônia Azul: recursos e preservação. *Revista USP*, n. 113, p. 7–26, 2017.

CHEN, X.-W.; LIN, X. Big data deep learning: challenges and perspectives. *IEEE Access*, IEEE, v. 2, p. 514–525, 2014.

CHEN, Z.; EAVANI, H.; CHEN, W.; LIU, Y.; WANG, W. Y. Few-shot NLG with pre-trained language model. *arXiv preprint arXiv:1904.09521*, 2019.

CHO, K.; MERRIËNBOER, B. V.; GULCEHRE, C.; BAHDANAU, D.; BOUGARES, F.; SCHWENK, H.; BENGIO, Y. Learning phrase representations using RNN encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.

COLIN, E.; GARDENT, C.; M'RABET, Y.; NARAYAN, S.; PEREZ-BELTRACHINI, L. The webnlg challenge: Generating text from Dbpedia data. In: *Proceedings of the 9th International Natural Language Generation conference*. [S.l.: s.n.], 2016. p. 163–167.

CONNEAU, A.; BAEVSKI, A.; COLLOBERT, R.; MOHAMED, A.; AULI, M. Unsupervised cross-lingual representation learning for speech recognition. *arXiv preprint arXiv:2006.13979*, 2020.

COSTA, B. H. e; GONÇALVES, J. M.; GONÇALVES, E. J. Un Ocean conference needs transparent and science-based leadership on ocean conservation. *Marine Policy*, Elsevier, v. 143, p. 105197, 2022.

DALE, R. Natural language generation: The commercial state of the art in 2020. *Natural Language Engineering*, Cambridge University Press, v. 26, n. 4, p. 481–487, 2020.

DAUMÉ III, H.; KNIGHT, K.; LANGKILDE, I.; MARCU, D.; YAMADA, K. The importance of lexicalized syntax models for natural language generation tasks. In: *Proceedings of the International Natural Language Generation Conference*. [S.l.: s.n.], 2002. p. 9–16.

DEEMTER, K. V.; THEUNE, M.; KRAHMER, E. Real versus template-based natural language generation: A false opposition? *Computational Linguistics*, MIT Press One Rogers Street, Cambridge, MA 02142-1209, USA journals-info . . . , v. 31, n. 1, p. 15–24, 2005.

DEVLIN, J.; CHANG, M.-W.; LEE, K.; TOUTANOVA, K. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

DIMITROMANOLAKI, A.; ANDROUTSOPOULOS, I. Learning to order facts for discourse planning in natural language generation. *arXiv preprint cs/0306062*, 2003.

DODDINGTON, G. Automatic evaluation of machine translation quality using n-gram co-occurrence statistics. In: *Proceedings of the Second International Conference on Human Language Technology Research*. [S.l.: s.n.], 2002. p. 138–145.

DÖRR, K. N. Mapping the field of algorithmic journalism. *Digital Journalism*, Taylor & Francis, 2015.

DUŠEK, O.; NOVIKOVA, J.; RIESER, V. Findings of the E2E NLG challenge. *arXiv preprint arXiv:1810.01170*, 2018.

DYK, D. A. V.; MENG, X.-L. The art of data augmentation. *Journal of Computational and Graphical Statistics*, Taylor & Francis, v. 10, n. 1, p. 1–50, 2001.

FEINER, S. K.; MCKEOWN, K. R. Comet: Generating coordinated multimedia explanations. In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. [S.l.: s.n.], 1991. p. 449–450.

FERNANDES, P. M. S. Community-based Sports Articles Generation Platform using NLG and Post-Editing. 2021.

FERREIRA, T. C.; LEE, C. van der; MILTENBURG, E. V.; KRAHMER, E. Neural data-to-text generation: A comparison between pipeline and end-to-end architectures. 2019.

FIRAT, F. Robot journalism. *The International Encyclopedia of Journalism Studies*, John Wiley & Sons, Inc. Hoboken, NJ, USA, p. 1–5, 2019.

FRENCH, R. M. The Turing Test: the first 50 years. *Trends in Cognitive Sciences*, Elsevier, v. 4, n. 3, p. 115–122, 2000.

FRISONI, G.; CARBONARO, A.; MORO, G.; ZAMMARCHI, A.; AVAGNANO, M. Nlg-Metricverse: An End-to-End Library for Evaluating Natural Language Generation. In: *Proceedings of the 29th International Conference on Computational Linguistics*. [S.l.: s.n.], 2022. p. 3465–3479.

FURTADO, S. d. F. D. Automated Journalism in Brazil: an Analysis of Three Robots on Twitter. *Brazilian Journalism Research*, v. 16, n. 3, p. 476–501, 2020.

GARDENT, C.; SHIMORINA, A.; NARAYAN, S.; PEREZ-BELTRACHINI, L. The WebNLG challenge: Generating text from RDF data. In: *Proceedings of the 10th International Conference on Natural Language Generation*. [S.l.: s.n.], 2017. p. 124–133.

GATT, A.; KRAHMER, E. Survey of the state of the art in natural language generation: Core tasks, applications and evaluation. *Journal of Artificial Intelligence Research*, v. 61, p. 65–170, 2018.

GRAEFE, A. Guide to automated journalism. 2016.

GRAEFE, A. Computational Campaign Coverage. 2017.

GRAEFE, A.; BOHLKEN, N. Automated journalism: A meta-analysis of readers' perceptions of human-written in comparison to automated news. *Media and Communication*, PRT, v. 8, n. 3, p. 50–59, 2020.

GRAEFE, A.; HAIM, M.; HAARMANN, B.; BROSIUS, H.-B. Readers' perception of computer-generated news: Credibility, expertise, and readability. *Journalism*, SAGE Publications Sage UK: London, England, v. 19, n. 5, p. 595–610, 2018.

HAQUE, M. U.; DHARMADASA, I.; SWORNA, Z. T.; RAJAPAKSE, R. N.; AHMAD, H. "I think this is the most disruptive technology": Exploring Sentiments of ChatGPT Early Adopters using Twitter Dsata. *arXiv preprint arXiv:2212.05856*, 2022.

HARTLEY, A.; PARIS, C. Automatic text generation for software development and use. *BENJAMINS TRANSLATION LIBRARY*, JOHN BENJAMINS BV, v. 18, p. 221–242, 1996.

HE, X.; DENG, L. Deep learning in natural language generation from images. In: *Deep learning in natural language processing*. [S.l.]: Springer, 2018. p. 289–307.

HEILBRON, M.; EHINGER, B.; HAGOORT, P.; LANGE, F. P. D. Tracking naturalistic linguistic predictions with deep neural language models. *arXiv preprint arXiv:1909.04400*, 2019.

HORACEK, H. *Building Natural Language Generation Systems*. [S.l.]: MIT Press One Rogers Street, Cambridge, MA 02142-1209, USA journals-info . . . , 2001.

JOSHI, A.; KALE, S.; CHANDEL, S.; PAL, D. K. Likert scale: Explored and explained. *British Journal of Applied Science & Technology*, SCIENCEDOMAIN International, v. 7, n. 4, p. 396, 2015.

JURASKA, J.; WALKER, M. Attention is indeed all you need: Semantically attention-guided decoding for data-to-text NLG. *arXiv preprint arXiv:2109.07043*, 2021.

KAELBLING, L. P.; LITTMAN, M. L.; MOORE, A. W. Reinforcement learning: A survey. *Journal of Artificial Intelligence Research*, v. 4, p. 237–285, 1996.

KIM, T.-Y.; BAE, S.-H.; AN, Y.-E. Design of smart home implementation within IoT natural language interface. *IEEE Access*, IEEE, v. 8, p. 84929–84949, 2020.

LANGKILDE, I.; KNIGHT, K. Generation that exploits corpus-based statistical knowledge. In: *COLING 1998 Volume 1: The 17th International Conference on Computational Linguistics*. [S.l.: s.n.], 1998.

LAVIE, A.; DENKOWSKI, M. J. The METEOR metric for automatic evaluation of machine translation. *Machine translation*, Springer, v. 23, n. 2, p. 105–115, 2009.

LEÃO, Z.; KIKUCHI, R. The Abrolhos reefs of Brazil. In: *Coastal Marine Ecosystems of Latin America*. [S.l.]: Springer, 2001. p. 83–96.

LI, H. Deep learning for natural language processing: advantages and challenges. *National Science Review*, 2017.

LIDDY, E. D. Natural language processing. 2001.

LIN, C.-Y.; OCH, F. Looking for a few good metrics: ROUGE and its evaluation. In: *Ntcir workshop*. [S.l.: s.n.], 2004.

LINDÉN, C.-G.; TUULONEN, H.; BÄCK, A.; DIAKOPOULOS, N.; GRANROTH-WILDING, M.; HAAPANEN, L.; LEPPÄNEN, L.; MELIN, M.; MORING, T.; MUNEZERO, M. et al. News automation: The rewards, risks and realities of 'machine journalism'. WAN-IFRA, 2019.

LIPTON, Z. C.; ELKAN, C.; NARAYANASWAMY, B. Thresholding classifiers to maximize F1 score. *arXiv preprint arXiv:1402.1892*, 2014.

MELLISH, C.; DALE, R. Evaluation in the context of natural language generation. *Computer Speech & Language*, Elsevier, v. 12, n. 4, p. 349–373, 1998.

MELLISH, C.; REITER, E.; LEVINE, J. NLG applications to technical documentation A view through IDAS. *Trends in Natural Language Generation An Artificial Intelligence Perspective*, 1996.

MIRTAHERI, S. M.; DINÇKTÜRK, M. E.; HOOSHMAND, S.; BOCHMANN, G. V.; JOURDAN, G.-V.; ONUT, I. V. A brief history of web crawlers. *arXiv preprint arXiv:1405.0749*, 2014.

MONTAL, T.; REICH, Z. I, robot. You, journalist. Who is the author? Authorship, bylines and full disclosure in automated journalism. *Digital journalism*, Taylor & Francis, v. 5, n. 7, p. 829–849, 2017.

MUTTON, A.; DRAS, M.; WAN, S.; DALE, R. GLEU: Automatic evaluation of sentence-level fluency. In: *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*. [S.l.: s.n.], 2007. p. 344–351.

NASUKAWA, T.; YI, J. Sentiment analysis: Capturing favorability using natural language processing. In: *Proceedings of the 2nd international conference on Knowledge capture*. [S.l.: s.n.], 2003. p. 70–77.

NESTERENKO, L. Building a system for stock news generation in Russian. In: *Proceedings of the 2nd international workshop on natural language generation and the semantic web (webnlg 2016)*. [S.l.: s.n.], 2016. p. 37–40.

NIE, F.; YAO, J.-G.; WANG, J.; PAN, R.; LIN, C.-Y. A simple recipe towards reducing hallucination in neural surface realisation. In: *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. [S.l.: s.n.], 2019. p. 2673–2679.

PAPINENI, K.; ROUKOS, S.; WARD, T.; ZHU, W.-J. Bleu: a method for automatic evaluation of machine translation. In: *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*. [S.l.: s.n.], 2002. p. 311–318.

PARMAR, S.; MESHRAM, M.; PARMAR, P.; PATEL, M.; DESAI, P. Smart hotel using intelligent chatbot: A review. *International Journal of Scientific Research in Computer Science, Engineering and Information Technology*, v. 5, n. 2, p. 823–829, 2019.

PEREIRA, J. C.; TEIXEIRA, A.; PINTO, J. S. Towards a hybrid NLG system for data2text in Portuguese. In: IEEE. *2015 10th Iberian Conference on Information Systems and Technologies (CISTI)*. [S.l.], 2015. p. 1–6.

PIROZELLI, P.; CASTRO, A. B. R.; OLIVEIRA, A. L. C. de; OLIVEIRA, A. S.; CAçãO, F. N.; SILVEIRA, I. C.; CAMPOS, J. G. M.; MOTHEO, L. C.; FIGUEIREDO, L. F.; PELLICER, L. F. A. O.; JOSé, M. A.; JOSé, M. M.; LIGABUE, P. de M.; GRAVA, R. S.; TAVARES, R. M.; MATOS, V. B.; SYM, Y. V.; COSTA, A. H. R.; BRANDãO, A. A. F.; MAUá, D. D.; COZMAN, F. G.; PERES, S. M. The BLue Amazon Brain (BLAB): A Modular Architecture of Services about the Brazilian Maritime Territory. *arXiv preprint arXiv:2209.07928*, 2022.

PU, A.; CHUNG, H. W.; PARIKH, A. P.; GEHRMANN, S.; SELLAM, T. Learning compact metrics for mt. *arXiv preprint arXiv:2110.06341*, 2021.

RAMOS-SOTO, A.; BUGARÍN, A.; BARRO, S. Fuzzy sets across the natural language generation pipeline. *Progress in Artificial Intelligence*, Springer, v. 5, n. 4, p. 261–276, 2016.

REITER, E. NLG vs. templates. *arXiv preprint cmp-lg/9504013*, 1995.

REITER, E. A structured review of the validity of BLEU. *Computational Linguistics*, MIT Press One Rogers Street, Cambridge, MA 02142-1209, USA journals-info . . . , v. 44, n. 3, p. 393–401, 2018.

REITER, E.; DALE, R. Building applied natural language generation systems. *Natural Language Engineering*, 2000.

REITER, E.; SRIPADA, S. Human variation and lexical choice. *Computational Linguistics*, MIT Press One Rogers Street, Cambridge, MA 02142-1209, USA journals-info . . . , v. 28, n. 4, p. 545–553, 2002.

RUDOLPH, J.; TAN, S.; TAN, S. ChatGPT: Bullshit spewer or the end of traditional assessments in higher education? *Journal of Applied Learning and Teaching*, v. 6, n. 1, 2023.

SAHATQIJA, K.; AJDARI, J.; ZENUNI, X.; RAUFI, B.; ISMAILI, F. Comparison between relational and NOSQL databases. In: IEEE. *2018 41st international convention on information and communication technology, electronics and microelectronics (MIPRO)*. [S.l.], 2018. p. 0216–0221.

SAI, A. B.; MOHANKUMAR, A. K.; KHAPRA, M. M. A survey of evaluation metrics used for NLG systems. *ACM Computing Surveys (CSUR)*, ACM New York, NY, v. 55, n. 2, p. 1–39, 2022.

SELLAM, T.; DAS, D.; PARIKH, A. P. Bleurt: Learning robust metrics for text generation. *arXiv preprint arXiv:2004.04696*, 2020.

SHUSTER, K.; XU, J.; KOMEILI, M.; JU, D.; SMITH, E. M.; ROLLER, S.; UNG, M.; CHEN, M.; ARORA, K.; LANE, J. et al. BlenderBot 3: a deployed conversational agent that continually learns to responsibly engage. *arXiv preprint arXiv:2208.03188*, 2022.

SRIPADA, S.; BURNETT, N.; TURNER, R.; MASTIN, J.; EVANS, D. A case study: NLG meeting weather industry demand for quality and quantity of textual weather forecasts. In: *Proceedings of the 8th International Natural Language Generation Conference (INLG)*. [S.l.: s.n.], 2014. p. 1–5.

SRIPADA, S. G.; REITER, E.; DAVY, I.; NILSSEN, K. Lessons from deploying NLG technology for marine weather forecast text generation. *WEATHER*, Citeseer, v. 5, p. 7, 2004.

STEDE, M. Lexicalization in natural language generation: A survey. *Artificial Intelligence Review*, Springer, v. 8, n. 4, p. 309–336, 1994.

SYM, Y. V.; CAMPOS, J. G. M.; COZMAN, F. G. BLAB Reporter: Automated journalism covering the Blue Amazon. *INLG 2022*, p. 1, 2022.

SYM, Y. V.; CAMPOS, J. G. M.; JOSÉ, M. M.; COZMAN, F. G. Comparing computational architectures for automated journalism. *arXiv preprint arXiv:2210.04107*, 2022.

TANNER, M. A.; WONG, W. H. The calculation of posterior distributions by data augmentation. *Journal of the American Statistical Association*, Taylor & Francis, v. 82, n. 398, p. 528–540, 1987.

TEIXEIRA, A. L. R.; CAMPOS, J.; CUNHA, R.; FERREIRA, T. C.; PAGANO, A.; COZMAN, F. DaMata: A robot-journalist covering the Brazilian Amazon deforestation. In: *Proceedings of the 13th International Conference on Natural Language Generation*. [S.l.: s.n.], 2020.

THAWANI, A.; PUJARA, J.; SZEKELY, P. A.; ILIEVSKI, F. Representing numbers in nlp: a survey and a vision. *arXiv preprint arXiv:2103.13136*, 2021.

THOMPSON, B.; POST, M. Automatic machine translation evaluation in many languages via zero-shot paraphrasing. *arXiv preprint arXiv:2004.14564*, 2020.

THOMSON, C.; REITER, E. A gold standard methodology for evaluating accuracy in data-to-text systems. *arXiv preprint arXiv:2011.03992*, 2020.

TORRIJOS, J. L. R. Semi-automated Journalism: Reinforcing Ethics to Make the Most of Artificial Intelligence for Writing News. *News Media Innovation Reconsidered: Ethics and Values in a Creative Reconstruction of Journalism*, Wiley Online Library, p. 124–137, 2021.

TOUSIGNANT, B. *A Hybrid Analysis of the State of Automated Journalism in Canada: Current Impact and Future Implications for Journalists and Newsrooms*. Tese (Doutorado) — Concordia University, 2020.

UCHENDU, A.; MA, Z.; LE, T.; ZHANG, R.; LEE, D. TURINGBENCH: A Benchmark Environment for Turing Test in the Age of Neural Text Generation. *arXiv preprint arXiv:2109.13296*, 2021.

VASWANI, A.; SHAZEER, N.; PARMAR, N.; USZKOREIT, J.; JONES, L.; GOMEZ, A. N.; KAISER, Ł.; POLOSUKHIN, I. Attention is all you need. *Advances in Neural Information Processing Systems*, v. 2017-Decem, n. Nips, p. 5999–6009, 2017. ISSN 10495258.

VASWANI, A.; SHAZEER, N.; PARMAR, N.; USZKOREIT, J.; JONES, L.; GOMEZ, A. N.; KAISER, Ł.; POLOSUKHIN, I. Attention is all you need. *Advances in Neural Information Processing Systems*, v. 30, 2017.

VIG, J.; BELINKOV, Y. Analyzing the structure of attention in a transformer language model. *arXiv preprint arXiv:1906.04284*, 2019.

WEISS, K.; KHOSHGOFTAAR, T. M.; WANG, D. A survey of transfer learning. *Journal of Big data*, SpringerOpen, v. 3, n. 1, p. 1–40, 2016.

WIESEBRON, M. Blue Amazon: thinking about the defence of the maritime territory. *Austral: Brazilian Journal of Strategy & International Relations*, v. 2, n. 3, p. 107–132, 2013.

XUE, L.; CONSTANT, N.; ROBERTS, A.; KALE, M.; AL-RFOU, R.; SIDDHANT, A.; BARUA, A.; RAFFEL, C. mT5: A massively multilingual pre-trained text-to-text transformer. *arXiv preprint arXiv:2010.11934*, 2020.

ZHANG, B.; YANG, J.; LIN, Q.; SU, J. Attention regularized sequence-to-sequence learning for e2e nlg challenge. *E2E NLG Challenge System Descriptions*, 2018.