Comparing Computational Architectures for Automated Journalism

Yan V. Sym[1,[0000−0002−0401−0586]], **João Gabriel M. Campos**[1,[0000−0002−1106−3694]], **Marcos M. José** [1,[0000−0003−4663−4386]], **Fabio G. Cozman** [1,[0000−0003−4077−4935]]

[1]Escola Politécnica, Universidade de São Paulo, São Paulo, Brazil

Abstract. *The majority of NLG systems have been designed following either a template-based or a pipeline-based architecture. Recent neural models for data-to-text generation have been proposed with an end-to-end deep learning flavor, which handles non-linguistic input in natural language without explicit intermediary representations. This study compares the most often employed methods for generating Brazilian Portuguese texts from structured data. Results suggest that explicit intermediate steps in the generation process produce better texts than the ones generated by neural end-to-end architectures, avoiding data hallucination while better generalizing to unseen inputs. Code and corpus are publicly available.* [1]

Keywords – Natural Language Generation, Automated Journalism, Blue Amazon

1. Introduction

Natural Language Generation (NLG) is a subfield 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 and 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].

Multiple successful examples of data-to-text systems can be found in weather forecasting [Sripada et al. 2004], financial and analytical reporting, industrial monitoring [Kim et al. 2020] and conversational agents. 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].

Traditionally, most data-to-text applications have been designed in a modular fashion as this facilitates reuse in different domains; going directly from input to output with rules has been simply too complex [Gatt and 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

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

approach, which embodies linguistic insights to convert data to text by applying a series of sequential steps.

The emergence of neural-based NLG systems in the 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.

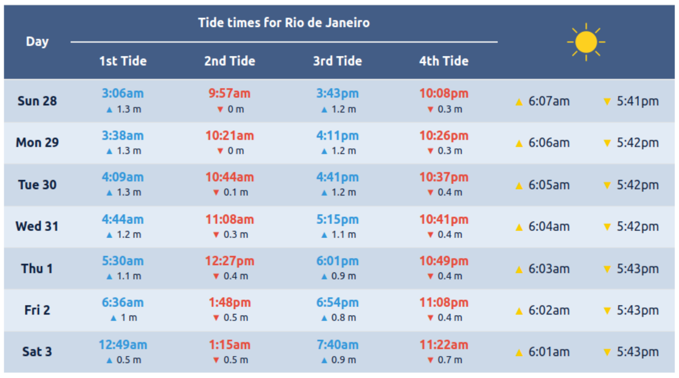
Even though it is technically feasible to use neural end-to-end methods in real world applications, this does not necessarily mean that they are superior to rule-based approaches in every scenario. Recent empirical studies have demonstrated that a combination of template and pipeline systems produce texts that are more appropriate than the neural-based approaches, which frequently hallucinate 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 reports. A modular model also has the advantage of allowing for auditing, while neural end-to-end approaches behave as black-boxes [Campos et al. 2020].

In this paper, we compare the three most frequently used architectures for automated journalism – template-based, pipeline-based and end-to-end neural models – using a common domain, the Blue Amazon. With an offshore area of 3.6 million square kilometers along the Brazilian coast, the Blue Amazon is Brazil's exclusive economic zone (EEZ); it is a oceanic region brimming with marine species and energy resources [Thompson and Muggah 2015]. Ocean monitoring, climate change and environmental sustainability are promising fields for automated journalism applications. The oceans are severely damaged environments, and if current trends continues, there will be disastrous consequences for the planet as it is essential to halt climate change, fostering economic growth and preserving biodiversity [e Costa et al. 2022]. Although connecting with public audiences in an approachable way typically requires coverage by trained human journalists, accurate and low latency information reports can be very helpful. There is a vast and ever-growing body of information about the oceans; clearly, society can benefit from a robot journalism system. To address this issue, we created our robot journalism application which combines different NLG approaches to generate daily reports about the Blue Amazon and publish them on Twitter. [2]

A corpus of verbalizations of non-linguistic data in Brazilian Portuguese 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. 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.

This main contributions of this work are the construction of a publicly available Brazilian Portuguese NLG dataset, a comparison between the three most frequently used automated journalism architectures and an application which combines different ap-

[2]`https://twitter.com/BLAB_Reporter`



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

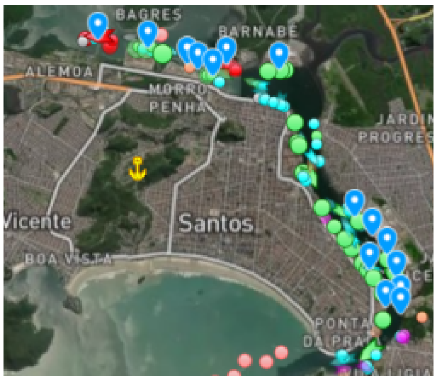


Figure 1. Left: tides chart for the Rio de Janeiro (RJ) city, taken from the Tides Chart website. Right: vessel positions near the Santos (SP) port on a given day. Taken from the Marine Traffic website.

proaches to publish daily reports about the Blue Amazon on Twitter. In Section 2, we present our Blue Amazon dataset for automated journalism, and in Section 3 we discuss our approach in building a template-based architecture. Also, in Section 4, we present and discuss our pipeline architecture with six sequential modules. In Section 5, we discuss the end-to-end architecture and utilize it by training four different neural networks to generate the output text. In Section 6 we present the main results of this work and in Section 7 we discuss the results by providing some qualitative analysis. Finally, we conclude in Section 8.

2. Non-linguistic Data about the Blue Amazon

The experiments presented in this work were run with a corpus of Brazilian Portuguese verbalizations for the Blue Amazon domain. We initially developed web crawlers which extracted daily information from publicly available sources, including weather, temperature, tides charts, earthquakes, vessel positioning and oil extraction. Weather data and tides charts are extracted through the Tides Chart website, 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 1 (left) shows an example of tides charts for the following week in Rio de Janeiro (RJ).

Vessel positioning is collected from the Marine Traffic website, 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. Figure 1 (right) shows an example of vessel positions near the Santos (SP) port on a given day. Real time data regarding earthquakes in the Brazilian coast are taken from the Seismological Center at the University of São Paulo, and information regarding oil extractinon is obtained from the Brazilian government portal. After the data are collected and cleaned, they are stored in the MongoDB database, a NoSQL document-oriented database program which provides more flexibility and scalability over relational databases when input data is constantly changing [Stonebraker 2010].

We created the corpus based on information collected during 90 consecutive days for 50 cities in the Brazilian coast, and then performed content selection for past time-series data using feedback from 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 300 texts extracted from the data. Syntactic and lexical patterns in the samples were used to produce a variety of target intent texts. Finally, intermediate representations in the 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");
WEATHER(condition="sunny",temperature="32°C");
EARTHQUAKE(magnitude="1.4 mR",depth="15km");
VESSELS IN PORT(quantity="350",trend="high",days max="30")
```

3. Template Architecture

Template-based data-to-text NLG systems directly translate non-linguistic input to linguistic surface structure by filling gaps in predefined template texts [Reiter 1995]. 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 design sentence planning modules like discourse ordering, sentence formation and lexicalization. However, the main advantage of template-based approaches happens in cases where good linguistic rules are not yet available or in highly specific conditions where only a few texts are possible, and thus there is no real benefit in having a highly complex NLG system [Smiley et al. 2018].

For example, a simple template-based system might start out from a semantic representation saying that a new earthquake with a magnitude of 1.7 mR and depth of 10km was detected by the Seismology Center at 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 at the University of São Paulo")
```

This intent-attribute-value input is then directly associated with a template using a rule-based approach, such as:

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

In this example, the gaps represented by [location], [magnitude], [depth] and [entity] are filled by looking up the relevant information in a table and the text will be generated only when a new earthquake is detected.

Some examples of template-based systems which publish daily reports on Twitter in Portuguese are Rosie from the Serenata de Amor operation, which identifies public funds expenses with discrepancies and indicates the reasons that lead it to believe they are suspicious, and Rui Barbot, which monitors stalled processes in the Supreme Federal Court of Brazil (STF) [Furtado 2020].

4. Pipeline Architecture

The pipeline architecture converts structured input data to output text in 6 sequential steps: *Content Selection, Discourse Ordering, Text Structuring, Lexicalization, Referring Expression Generation and Textual Realization* [Horacek 2001]. Figure 2 shows the pipeline

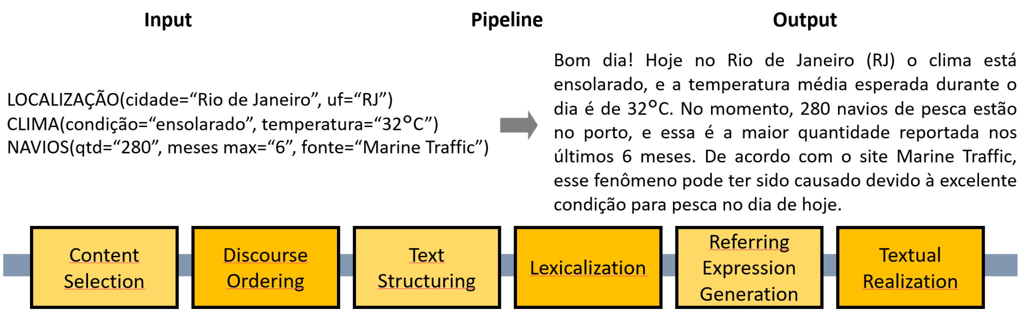

Input
Pipeline
Output
LOCALIZAÇÃO(cidade="Rio de Janeiro", uf="RJ")
CLIMA(condição="ensolarado", temperatura="32°C")
NAVIOS(qtd="280", meses max="6", fonte="Marine Traffic")
Bom dia! Hoje no Rio de Janeiro (RJ) o clima está ensolarado, e a temperatura média esperada durante o dia é de 32°C. No momento, 280 navios de pesca estão no porto, e essa é a maior quantidade reportada nos últimos 6 meses. De acordo com o site Marine Traffic, esse fenômeno pode ter sido causado devido à excelente condição para pesca no dia de hoje.
Content Selection
Discourse Ordering
Text Structuring
Lexicalization
Referring Expression Generation
Textual Realization

Figure 2. Pipeline architecture with an example of a generated text for our automated journalism application.

architecture steps with an example in the case of our automated journalism application. The system receives as input information regarding location, weather and vessels, and outputs the following text translated to English:

Good Morning! Today in Rio de Janeiro (RJ) the weather is sunny, and the average temperature expected during the day is 32°C. Currently, 280 fishing vessels are in 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 due to the excellent conditions for fishing today.

Content Selection In the first step of the pipeline architecture, the Content Selection module decides what information is important to be conveyed to the target audience and verbalized in the text. Since content determination precedes language generation, template-based systems can treat it in the exact same ways as 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 et al. 2005]. This step often requires the assistance of a domain expert to understand what information is relevant in the given context and how to group them in intent messages. The following text is an example of the content selection module output for our application:

```
LOCATION(city="Santos",uf="SP",timestamp="Jan 15, 2022");
WEATHER(condition="sunny",temperature="32°C");
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 has been selected, the next step of the pipeline is to determine the order in which the intent messages should be verbalized to enhance reader comprehension [Heilbron et al. 2019]. Although some authors have had success with machine learning solutions to order facts for discourse planning [Dimitromanolaki and 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 and Walker 2021], most applications utilize explicit content text structuring.

For the case of our application, 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 lexicalization step aims to find the proper word and phrases to express the content in each sentence. This is performed by adding words, phrases or word patterns to a language's vocabulary to inflect words based on their grammatical use (tense, number, case and gender, for example) and verbalize the intents [Stede 1994]. For example, we might have as intent-attribute-value input:

```
LOCATION(city="Rio de Janeiro",uf="RJ",timestamp="Jan 15, 2022")
WEATHER(condition="cloudy",temperature="28°C",max_since_days="10")
```

And generates the following output texts in Portuguese, and translated to English:

Hoje, dia 15 de Janeiro de 2022, o clima é nublado no Rio de Janeiro (RJ). A temperatura média esperada é de 28ºC, e esta é a maior temperatura dos últimos 10 dias.

(Today, January 15th, 2022, the weather is cloudy in Rio de Janeiro (RJ). The average expected temperature is 28ºC, and this is the highest temperature in the last 10 days.)

Referring Expression Generation In order to replace entity tags throughout the template, this module aims to automatically generate 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.).

Textual Realization The last step of the pipeline approach performs the remaining adjustments to transform intermediate machine representations into text. Detokenization, contractions, nominal and verbal are performed in order to make the content grammatically consistent. For our robot-journalism application, this step applies a final layer of textual manipulation, to make the content more appealing to the target audience. The resulting texts are published every day using Twitter's API.

An example of pipeline-based natural language application which publishes daily reports on Twitter in Portuguese is DaMata, a robot-journalist system covering the Brazilian Amazon deforestation [Teixeira et al. 2020].

5. End-to-End Architecture

End-to-end architectures for natural language generation recently gained popularity due to the massive amount of data and computational power available [Chen and Lin 2014]. Given enough labeled data, it does become possible go 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 nenetworks, convolutional

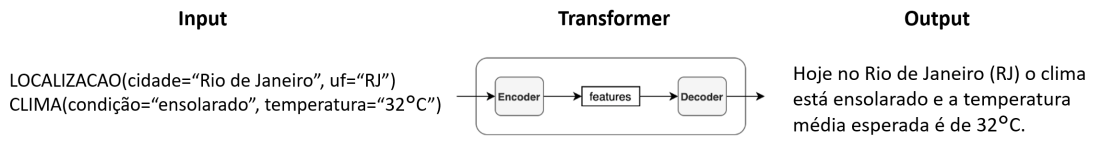

Input
Transformer
Output
LOCALIZACAO(cidade="Rio de Janeiro", uf="RJ")
CLIMA(condição="ensolarado", temperatura="32°C")
Encoder
features
Decoder
Hoje no Rio de Janeiro (RJ) o clima está ensolarado e a temperatura média esperada é de 32°C.

Figure 3. Example of a generated text for our automated journalism application using transformers.

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 and image captions [He and Deng 2018].

Our goal here was to learn a direct mapping from a intent-attribute-value input text to a human-readable Portuguese text. We tested four text-to-text transformer-based models: Bart [Lewis et al. 2019], T5 [Xue et al. 2020], Blenderbot [Shuster et al. 2022] and GPT2 [Radford et al. 2019]. While GPT2 utilizes a decoder only module, the first three models use a encoder-decoder scheme [Cho et al. 2014]. 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 3 shows an example of this process in the context of our automated journalism application: the system receives the text sequence: `Location(city= "Rio de Janeiro", state="RJ"); Weather(climate="sunny", temperature="32°C")` and outputs the following text, translated to English: *Today in Rio de Janeiro (RJ), the weather is sunny, and the average temperature expected during the day is 32ºC.*

For the experiments in this work, the dataset was randomly split into training (60%), validation (20%) and testing (20%). The models were trained on Google Colab using a NVIDIA Tesla K80 GPU for a maximum of 100 epochs with a learning rate of 1e-5 and an early stopping criteria of 3 epochs.

6. Comparing Architectures: Results

The results of the three different automated journalism architectures compared in this work were evaluated by averaging the opinion of 5 domain experts in a sample of 100 texts present in the test set. For each pair of input data and output text, the participants rated the trials 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?"*) of the Brazilian Portuguese output text in 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 four different commonly used natural language generation metrics: Bleu, [Papineni et al. 2002], Gleu [Mutton et al. 2007], Rouge [Lin and Och 2004] and Meteor [Lavie and Denkowski 2009]. Table 1 depicts the results of both automatic and human evaluations.

7. Discussion

The results presented in Table 1 show that the template-based architecture outperformed all the others in both fluency and semantics. As shown in the example of table 2, this

Table 1. Comparing the automated journalism architectures on the Blue Amazon dataset.

Architecture	Bleu	Gleu	Rouge	Meteor	Fluency	Semantics	Lexical Variety
Template	-	-	-	-	**4.8**	**4.8**	3.2
Pipeline	-	-	-	-	4.5	4.6	**4.4**
End-to-end (Bart)	**47.9**	**47.2**	57.7	71.6	3.8	3.5	3.7
End-to-end (T5)	43.6	46.5	**58.2**	**73.2**	3.5	3.3	3.7
End-to-end (Blenderbot)	38.4	36.1	52.9	60.3	3.2	3.2	3.1
End-to-end (GPT2)	19.3	18.4	17.8	19.8	2.1	1.9	2.2

happens because the output texts are very repetitive and the architecture fails to provide lexical variety. This is a known limitation of the approach [Pereira et al. 2015], and after the results obtained in this work we opted to apply the template-based architecture only for sensitive and edge case 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 these metrics and received 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 conveyed, the pipeline-based architecture is more interesting than the template-based approach because it provides for more lexical variety to the target audience. It also has the advantage of not hallucinating data, unlike the novel end-to-end approaches.

Results from the neural end-to-end architecture show that all the four tested transformer-based models scored significantly less in all the human quantitative metrics. This happens because deep learning methods for NLG often hallucinates data and does not convey all the meaningful information in the input, as shown in table 2. This approach also makes 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, there are no domain experts to provide linguistic insights, which is not the case for our application.

As for the automatic metrics, the Bart model outperformed the other neural end-to-end models in both Bleu and Gleu, while the T5 achieved the best Rouge and Meteor scores. While Blenderbot obtained average scores, GPT2 obtained the lowest scores overall for all the purposed metrics. Works in simmilar contexts have reported BLEU scores ranging from 45% to 65% [Dušek et al. 2018]. Table 2 show 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 more lexical variety. The neural-based approach sometimes hallucinates both data and content, and also makes both lexical and semantic mistakes. Unlike the other 3 models, GPT2 utilizes a decoder only module, and the result was that it failed to complete long sentences and properly transform input data into coherent output text.

Table 2. 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="26°C", 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="33°C", 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="25°C"); 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="35°C", 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="25°C", 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.

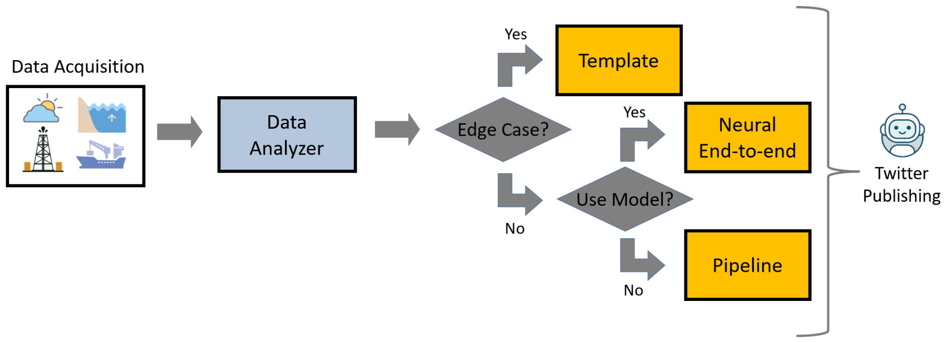


Figure 4. Automated journalism system architecture developed to publish daily reports about the Blue Amazon.

8. Conclusion

This paper compared data-to-text natural language generation architectures in the context of automated journalism for the Blue Amazon domain. We also created a publicly available dataset and an automated journalism system which publishes daily Brazilian Portuguese reports on Twitter by collecting, storing and analyzing information from multiple sources. Figure 4 shows the current architecture of our automated journalism application. After the data is collected and stored on our system's database, a rule-based data analyzer module decides whether the content is critical. If that is the case, we opted to use the Template approach to communicate the message and to avoid the risk of generating texts with poor fluency or semantics. If the data analyzer module decides that the information is not critical, results show that the pipeline approach produces better texts than neural end-to-end approaches, by avoiding data hallucination while also generalizing better to unseen inputs. For our application, we decided to implement a Boolean input parameter responsible for choosing between the pipeline and the neural end-to-end approaches. We have scheduled a weekly retraining of the neural end-to-end models to make sure that they will continue to perform as expected for new inputs, thus avoiding the risk of data drift.

In the future, we plan to add more sources of information to our application, such as real time reporting of illegal fishing activities and other natural disasters. We also plan to use different neural end-to-end methods at each step of the pipeline architecture, and to replace numbers with their equivalent text to validate whether better overall results can be achieved. We plan to experiment with text summarization architectures to outline public news related to the Blue Amazon in short tweets.

9. Acknowledgements

We would like to thank the Center for Artificial Intelligence (C4AI: www.c4ai.inova.usp), supported by São Paulo Research Foundation (FAPESP grant #2019/07665-4) and IBM Corporation. This research was also partially supported by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES), Finance Code 001. M. M. José, has been supported by the Itaú Scholarship Program (PBI), run by the Data Science Center (C2D) of the Escola Politécnica da Universidade de São Paulo. F. G. Cozman is partially supported by the National Council for Scientific and Technological Development of Brazil (CNPq) grant #312180/2018-7. Any opinions, findings, and conclusions expressed in this manuscript are those of the authors and do not necessarily reflect the views, official policy, or position of the financiers.

References

Campos, J., Teixeira, A., Ferreira, T., Cozman, F., and Pagano, A. (2020). Towards Fully Automated News Reporting in Brazilian Portuguese. In *Anais do XVII Encontro Nacional de Inteligência Artificial e Computacional*, pages 543–554. SBC.

Chen, X.-W. and Lin, X. (2014). Big data deep learning: challenges and perspectives. *IEEE access*, 2:514–525.

Cho, K., Van Merriënboer, B., Bahdanau, D., and Bengio, Y. (2014). On the properties of neural machine translation: Encoder-decoder approaches. *arXiv preprint arXiv:1409.1259*.

Deemter, K. V., Theune, M., and Krahmer, E. (2005). Real versus template-based natural language generation: A false opposition? *Computational linguistics*, 31(1):15–24.

Dimitromanolaki, A. and Androutsopoulos, I. (2003). Learning to order facts for discourse planning in natural language generation. *arXiv preprint cs/0306062*.

Dušek, O., Novikova, J., and Rieser, V. (2018). Findings of the e2e nlg challenge. *arXiv preprint arXiv:1810.01170*.

e Costa, B. H., Gonçalves, J. M., and Gonçalves, E. J. (2022). Un Ocean Conference needs transparent and science-based leadership on ocean conservation. *Marine Policy*, 143:105197.

Ferreira, T. C., van der Lee, C., Van Miltenburg, E., and Krahmer, E. (2019). Neural data-to-text generation: A comparison between pipeline and end-to-end architectures.

Furtado, S. d. F. D. (2020). Automated journalism in brazil: an analysis of three robots on Twitter. *Brazilian journalism research*, 16(3):476–501.

Gatt, A. and Krahmer, E. (2018). Survey of the state of the art in natural language generation: Core tasks, applications and evaluation. *Journal of Artificial Intelligence Research*, 61:65–170.

Graefe, A. (2016). Guide to automated journalism.

He, X. and Deng, L. (2018). Deep learning in natural language generation from images. In *Deep learning in natural language processing*, pages 289–307. Springer.

Heilbron, M., Ehinger, B., Hagoort, P., and De Lange, F. P. (2019). Tracking naturalistic linguistic predictions with deep neural language models. *arXiv preprint arXiv:1909.04400*.

Horacek, H. (2001). Building natural language generation systems.

Joshi, A., Kale, S., Chandel, S., and Pal, D. K. (2015). Likert scale: Explored and explained. *British journal of applied science & technology*, 7(4):396.

Juraska, J. and Walker, M. (2021). Attention is indeed all you need: Semantically attention-guided decoding for data-to-text NLG. *arXiv preprint arXiv:2109.07043*.

Kim, T.-Y., Bae, S.-H., and An, Y.-E. (2020). Design of smart home implementation within IoT natural language interface. *IEEE Access*, 8:84929–84949.

Lavie, A. and Denkowski, M. J. (2009). The METEOR metric for automatic evaluation of machine translation. *Machine translation*, 23(2):105–115.

Lewis, M., Liu, Y., Goyal, N., Ghazvininejad, M., Mohamed, A., Levy, O., Stoyanov, V., and Zettlemoyer, L. (2019). Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension.

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

Liddy, E. D. (2001). Natural language processing.

Lin, C.-Y. and Och, F. (2004). Looking for a few good metrics: ROUGE and its evaluation. In *Ntcir workshop*.

Mutton, A., Dras, M., Wan, S., and Dale, R. (2007). GLEU: Automatic evaluation of sentence-level fluency. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 344–351.

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

Pereira, J. C., Teixeira, A., and Pinto, J. S. (2015). Towards a hybrid NLG system for data2text in Portuguese. In *2015 10th Iberian Conference on Information Systems and Technologies (CISTI)*, pages 1–6. IEEE.

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., Sutskever, I., et al. (2019). Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Reiter, E. (1995). Nlg vs. templates. *arXiv preprint cmp-lg/9504013*.

Reiter, E. and Dale, R. (2000). Building Applied Natural Language Generation Systems. *Natural Language Engineering*.

Shuster, K., Xu, J., Komeili, M., Ju, D., Smith, E. M., Roller, S., Ung, M., Chen, M., Arora, K., Lane, J., et al. (2022). BlenderBot 3: a deployed conversational agent that continually learns to responsibly engage. *arXiv preprint arXiv:2208.03188*.

Smiley, C., Davoodi, E., Song, D., and Schilder, F. (2018). The E2E NLG challenge: A tale of two systems. In *Proceedings of the 11th International Conference on Natural Language Generation*, pages 472–477.

Sripada, S. G., Reiter, E., Davy, I., and Nilssen, K. (2004). Lessons from deploying NLG technology for marine weather forecast text generation. *WEATHER*, 5:7.

Stede, M. (1994). Lexicalization in natural language generation: A survey. *Artificial Intelligence Review*, 8(4):309–336.

Stonebraker, M. (2010). Sql databases v. nosql databases. *Communications of the ACM*, 53(4):10–11.

Teixeira, A. L. R., Campos, J., Cunha, R., Ferreira, T. C., Pagano, A., and Cozman, F. (2020). DaMata: A robot-journalist covering the Brazilian Amazon deforestation. In *Proceedings of the 13th International Conference on Natural Language Generation*.

Thompson, N. and Muggah, R. (2015). The Blue Amazon. *Foreign affairs*, 11.

Xue, L., Constant, N., Roberts, A., Kale, M., Al-Rfou, R., Siddhant, A., Barua, A., and Raffel, C. (2020). mt5: A massively multilingual pre-trained text-to-text transformer. *arXiv preprint arXiv:2010.11934*.