2021 IEEE 23rd Int Conf on High Performance Computing & Communications; 7th Int Conf on Data Science & Systems; 19th Int Conf on Smart City; 7th Int Conf on Dependability in Sensor, Cloud & Big Data Systems & Application

2021 IEEE 23rd Int Conf on High Performance Computing & Communications; 7th Int Conf on Data Science & Systems; 19th Int Conf on Smart City; 7th Int Conf on Dependability in Sensor, Cloud & Big Data Systems & Application (HPCC/DSS/SmartCity/DependSys) | 978-1-6654-9457-1/21/$31.00 ©2021 IEEE | DOI: 10.1109/HPCC-DSS-SMARTCITY-DEPENDSYS53884.2021.00334

A geometrical deep learning model for the lexicalisation of "unseen" RDF triples

Marco Cremaschi, Simone Saleri, Andrea Maurino
Department of Computer Science, Systems and Communication
University of Milano - Bicocca
Milano, Italy
marco.cremaschi@unimib.it, s.saleri@campus.unimib.it, andrea.maurino@unimib.it

***Abstract*—A considerable amount of data, presented in a structured form, is available on the Web nowadays. For the informational content of such data to be made accessible and understandable to users, its translation into text is preferable. This task is named "data-to-text generation" in the state-of-the-art, and it is an instance of the Natural Language Generation. In order to generate some valuable text from data, also known as lexicalisation, some approaches have begun to consider the Resource Description Format (RDF) data present within the Knowledge Graphs. In this context, it is possible to identify two main categories of lexicalisation approaches that use neural networks: pipeline and end-to-end. The former has better performances but is more complex to adapt. The latter, the end-to-end systems, has much simpler architectures but is less precise. In this work, in order to get the best from the two categories, we propose a new hybrid approach, TripleEnc, which, thanks to the concept of vector similarity between RDF triples, identifies the best approach for lexicalisation. Empirical comparisons demonstrate that the novel approach improves the quality of the generated text.**

***Keywords*-Natural Language Generation; RDF triples; Lexicalisation**

I. Introduction

Humans and machines do not speak the same language. What is easy for a human to understand can be hardly understandable by a computer and vice-versa. Machines work well with **data** such as:

⟨Buzz Aldrin | mission | Apollo 11⟩
⟨Buzz Aldrin | birthDate | 1930-01-20⟩
⟨Buzz Aldrin | birthPlace | Glen Ridge, New Jersey⟩
⟨Glen Ridge, New Jersey | country | United States⟩ (1)

Whilst humans prefer **text** like:

Buzz Aldrin,
born on January 20, 1930
at Glen Ridge in New Jersey, USA
was a crew member of Apollo 11. (2)

Therefore, it is necessary to find a technique to switch from one form to the other, i.e., mapping **data** to their respective **textual** representation. This task is named "data-to-text generation" in the state-of-the-art, and it is an instance of Natural Language Generation (NLG). NLG is a sub-filed of artificial intelligence and computational linguistics that is concerned with the construction of computer systems that can produce understandable texts in English or other human languages from some underlying non-linguistic representation of information [1].

The ability to transform **data** into an understandable **text** has many various applications: for instance, to give directions during emergencies when reading a screen is not comfortable or to generate breaking news in no time. Concerning this last example, a series of initiatives have taken place in journalism to develop algorithms capable of generating news under human supervision (i.e., Automated Journalism). Associated Press, Forbes, Los Angeles Times, and ProPublica are some of the companies that use such systems [2]. Fancier applications include personalised match reports or museum labels that change depending on the kind of audience[1]. The interest in this field is also certified by the birth of various commercial solutions; in 2016 there were 13 companies covering this field [3] (e.g., AutomatedInsights[2], NarrativeScience[3], AX NLG[4], and Arria[5]).

In order to generate valuable text, some approaches considering data contained in the Knowledge Graphs (KGs) have recently been introduced in literature [4], [5]. KGs are graph-based representations of concepts and their semantic relationships; often described using RDF, they are capable of representing knowledge on a certain topic. RDF is a semantic description language based on a set of triples of type ⟨**subject** | **predicate** | **object**⟩, where the subject is a resource, a concept or an entity, the predicate is a property describing the relation between the subject and the object, and the object is a literal, a concept, or another entity (e.g., ⟨subject: Buzz Aldrin | predicate: mission | object: Apollo 11⟩). Despite RDF being a powerful language, it can be challenging for non-experts to understand its meaning, as shown in the examples (1); for this reason, research in **RDF-**

[1]https://www.arria.com/use-cases-by-industry/
[2]https://automatedinsights.com/
[3]https://narrativescience.com/
[4]https://cockpit.ax-semantics.com/login
[5]https://www.arria.com/

978-1-6654-9457-1/21/$31.00 ©2021 IEEE
DOI 10.1109/HPCC-DSS-SmartCity-DependSys53884.2021.00334

to-text, though recently developed, is steadily growing and is aimed at efficiently **translating** information provided in RDF format into English text.

The data-to-text generation, as well as RDF-to-text generation, involves several sub-problems/tasks such as: i) content determination (select the data to lexicalise), ii) text structuring (determine the order of presentation of the information), iii) sentence aggregation (select the information to present), iv) lexicalisation (find the right word to express information), v) referring expression generation (select the word to identify domain), vi) linguistic realisation (combining words and phrases into well-formed sentences) [6], [1].

Many approaches in this context have been proposed for the last twenty years and can be divided into two main categories: **pipeline** and **end-to-end** systems [7]. The former have better performances but are more complex: they generally have a complex architecture composed of several modules, each treating one of the tasks defined above [6]; this kind of architecture is prone to error propagation, however, it is less susceptible to hallucinations, which happen when the system adds information that is not present in the input (basically, when they say things that are not true) [7]. The latter, the **end-to-end** systems, have much simpler architectures but are less precise: they are more prone to hallucinations, omissions and repetitions. Nevertheless, they excel in some tasks such as discourse ordering and text structuring [7]. Recent end-to-end approaches such as T5[6] [8] have closed the gap between the two architectures, surpassing pipelines in many tasks and showing excellent generalisation capabilities but still suffering from problems such as hallucinations, omissions and repetitions. On the other hand, pipeline systems like DualEnc [9] have demonstrated the importance of relations between the triples to create semantically correct sentences, which can be represented using graph structures such as **KGs**. These systems showed superior capabilities on *seen* domains (*seen* domains represent entities encountered during the training stage of the approach/system, conversely *unseen* domains represent those elements that the approach/system have never encountered before) but still suffer from the inability to generalise and describe the *unseen* domains. A system affected by these problems is unfaithful, therefore, utterly useless in a real-world scenario. Our **goal** is to face these problems with a strategy that **combines pipeline and end-to-end systems** so that we take the best from both worlds and the advantages are summed up.

In this work, we present TripleEnc, a **new hybrid approach** that starts from an *unseen* triple as input and identifies the most similar *seen* triple through the use of embeddings and vector similarity [10]. The *seen* triple is then lexicalised through the state-of-the-art pipeline approach (DualEnc [9]) appropriately modified. The obtained text is then updated by replacement with the subject and object initially present in the *unseen* triple. If the similarity between the *seen* and the *unseen* triple does not reach a certain threshold, our solution uses the state-of-the-art end-to-end approach (T5), which is based on a pre-trained language model. The generated text is statistically more correct, given the ability of T5 to lexicalise *unseen* triples.

We have fine-tuned and evaluated our approach on the WebNLG dataset [4], a widely used benchmark for RDF-to-text generation. Firstly, we have conducted a series of measurements to determine the proper similarity threshold, below which the substitutions were discarded in favour of the pre-trained language model approach; then, we have measured the multi-BLEU score of our model with the official WebNLG tools[7] and compared the results with the original unmodified baseline model. Lastly, we have conducted a human evaluation to measure to which extent humans accept our lexicalisations. Results on this dataset show that our model improves the quality of the generated text on the *unseen* domain. Results also reveal, as expected, that BLEU is not well suited as a unique measurement criterion: a human evaluation is still needed.

Ultimately, the main contributions of this paper are:

- a new RDF-to-text hybrid approach for the generation of natural language starting from seen or unseen triples;
- a systematic analysis of state-of-the-art to identify the strengths and weaknesses of the various RDF-to-text approaches.

The remainder of the paper is organised as follow: in Section II we propose a brief description about the problem of generating text from RDF triples. Section III describes the approach used for the evaluation of the new approach. In Section IV we contextualise this work through an analysis of the state-of-the-art. Our model is described in the Section V, while the experiments are analysed in the Section VI. Finally, in Section VII, we outline the future direction of research.

II. Problem statement

As introduced, in this work we focus on text generation from RDF data. The input is a set of RDF triples, denoted as $S = \{t_1, t_2, ...t_n\}$ where t_i is a triple $\langle s \mid p \mid o \rangle$ consisting of a subject, a predicate, and an object (1). The set of triples can be represented by a graph, as shown in Figure 1. The aim is to generate a natural language descriptive text $Y = \{w_1, w_2, ...w_t\}$ that represents the correct and concise semantics of entities and their relationships in the given RDF triple inputs [11] (2).

III. Evaluation criteria

It is important to set criteria to compare different outputs and evaluate the performances of different NLG systems.

[6]https://github.com/google-research/text-to-text-transfer-transformer

[7]https://github.com/WebNLG/GenerationEval

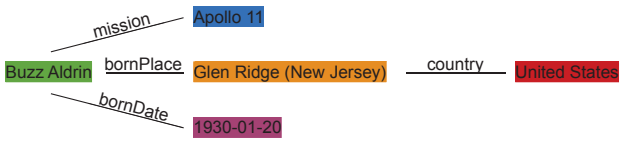

Figure 1. Graph representation of triples (1).

Over the years, many automatic evaluation metrics have been proposed for different purposes. The **Bilingual Evaluation Understudy (BLEU)** [12] was initially proposed for development purposes but became a de-facto standard for comparing models. It is based on the number of n-grams that appears both in the input and the output. This metric has been often criticised because of its flaws [13], such as not taking synonyms into account and not guaranteeing neither the grammatical correctness nor the faithfulness of the generated sequence, as shown in the example (5). During the years, other metrics have been proposed to target these problems, including **METEOR** [14] and **TER** [15]. Despite being very useful for automatic comparisons, they still are not even comparable to the **human evaluation**[8]. Recently, a new promising metric was proposed; it is a set of tests named **GEM Benchmark**[9], which also includes a human evaluation stage, aiming at defining a replicable standard for human evaluations.

IV. Related work

As mentioned in the introduction (Section I), it is possible to identify mainly three categories of approaches to NLG: **pipeline**, **end-to-end** and **hybrid**. The first is more traditional, whereas the others are more recent and owe their existence to the rise of Deep Learning (DL).

A. Pipeline VS end-to-end VS hybrid architectures

The NLG problem was initially addressed using **pipeline** architectures: these architectures are composed of many modules, each addressing one or more NLG tasks. Different pipelines may differ in the number of modules, but they have to face all six NLG tasks to produce good results. After the rise of DL techniques, there has been much interest in an alternative architecture, known as "end-to-end". An **end-to-end** architecture considers all NLG tasks within a "black-box"; it has a much simpler structure, but it is also much harder to tweak. Pipeline architectures are affected mainly by three problems: they are more costly, complex, and subject to error propagation. On the other hand, end-to-end methods can be trained with a massive amount of data and texts freely available on the Web, thus being very interesting, but their performances significantly drop in the *unseen* domains compared with pipelines [7]. Pure end-to-end architectures have shown their limits, being heavily affected by the drop of performances in the *unseen* domains and by hallucinations, which happen when the system adds pieces of information to the final output that are not present in the input [16]. At the same time, some end-to-end architectures have shown state-of-the-art results in a few specific NLG tasks [7]. For these reasons, most recent approaches do not belong to a specific category. Instead, they are **hybrid**. They somehow lie between the two categories: some additional steps were added to the canonic encoder-decoder architecture trying to fill the gap between the two and preserving intra-triple and inter-triple relations [17], improving the planning stage [9], hence the final output in the *unseen* domains.

B. Neural networks systems.

In the context of RDF-to-text, models that transform the RDF triples into a sequence of words may lose important higher-order information [5] because they do not directly exploit the input structure but rather rely on a separate linearisation step [17]. For this reason, **graph-based triple encoder models** such as GTR-LSTM [5] were to maintain the structure of RDF triples in the form of a small Knowledge Graph (KG), preserving as much information as possible, such as the relationships between different triples. In order to capture more complex structure information, a graph encoder based on a new variant of a Graph Convolutional Network (GCN) [18] was proposed in late 2018 [17] and inspired other researchers to propose new graph-to-sequence models. One of them is called Graph2Seq [19] and is characterised by the usage of an Attention Mechanism [20] to manage large graphs. Compared to sequence-to-sequence ones, graph-to-sequence architectures contain much more information, increasing the gap between the graph source and the linearised target, increasing the difficulty of establishing alignments between them. Due to this issue, a **dual encoder** model was proposed, namely DualEnc [9], aiming at **reducing the gap between the encoder and the decoder**. This model adds an intermediate stage to perform the "text structuring" task and produces significant improvements, reaching the state-of-the-art at the time of writing [9]. An alternative and recent way to approach the NLG is represented by transformers [21], in particular by those who rely on transfer learning such as GPT-3 [22] and T5 [8]. **Transfer learning** is a technique consisting of pre-training on a large general corpus of text, then fine-tuning on a specific task [22]. These models have shown **greater generalisation capabilities** thanks to the "general knowledge" they obtain with extensive datasets, which typically are much larger if compared with the datasets of specific tasks (e.g., the WebNLG one). Since GPT-3 has been acquired by Microsoft[10], The state-of-the-

[8]https://ehudreiter.com/2017/05/03/metrics-nlg-evaluation/
[9]https://gem-benchmark.com/

[10]Blog article about Microsoft's acquisition of the GPT-3: https://blogs.microsoft.com/blog/2020/09/22/microsoft-teams-up-with-openai-to-exclusively-license-gpt-3-language-model/

art for this category is represented by T5 [8].

V. Proposed Approach

All of the systems analysed in the previous Section suffer from the above-mentioned severe problems, including an essential drop of performances in the *unseen* domains; even DualEnc has a considerable drop, from 64.42 (*seen* domain) to 38.23 (*unseen* domain) points of BLEU scale [12] (automatic evaluation scale from 0 to 100, the higher, the better) [9]. Since we have detected that one of the most challenging problems for a data-to-text model is being able to describe *unseen* elements without falling into **hallucinations**, we are focusing on improving this ability, also known as **generalisation**. In an effort to find the solution, we came up with an intuition: it is possible to hypothesise that similar concepts should be lexicalised following similar rules, as they share most of the predicates. This is analogous to what children do when they learn to talk [23].

To clarify this scenario, let's consider this triple:

$$\langle \text{Fuffy (dog)} \mid \text{walk} \mid \text{ground} \rangle \tag{3}$$

The triple (3) is lexicalised by the DualEnc model in this way:

$$\text{Fuffy (dog)'s ground is the ground of Fuffy (dog).} \tag{4}$$

The above lexicalisation is wrong because the elements in the triple are of type *unseen*. As depicted, the *unseen* elements are those elements that the system has never encountered, and therefore the system does not know how to produce the textual representation (lexicalisation). The actual state-of-the-art model, DualEnc, has been trained with the WebNLG 2017 dataset[11] which considers the following *seen* categories: Airport, Astronaut, Building, City, ComicsCharacter, Food, Monument, SportsTeam, University, WrittenWork. If we consider the intuition described above, we can try to identify among the triples of the *seen* type (the triples inside the WebNLG 2017 train dataset), the one that is most similar to the triple about the dog, and then adapt the proposed lexicalisation.

Within the training dataset, the triple closest to the triple (3) may be:

$$\langle \text{Buzz Aldrin} \mid \text{stepOn} \mid \text{Moon} \rangle \tag{5}$$

This triple would be lexicalised as follows:

$$\text{Buzz Aldrin has stepped on the Moon.} \tag{6}$$

Through a substitution of the subject and the object in the sentence (6), we can obtain the following sentence:

$$\text{Fuffy (dog) has stepped on the ground.} \tag{7}$$

The sentence (7) is more semantically and syntactically correct with respect to the sentence (4); therefore, it is more concise and fluent, so it will be preferred by a human.

Unfortunately, given the limitations of the dataset, this hypothetical extreme case never happens because it considers three similar elements. What happens in a real scenario is that the system already knows one or two similar elements of the triple. In order to extend our idea to limited datasets, we propose a strategy that weights each element and assigns a score to each triple, ultimately finding the proper substitute. Empirical experiments highlighted the importance of the predicate at producing templates and the minor importance of the object. For these reasons, the score is assigned as a weighted average, defined as follows. Given a triple t and a substitute triple denoted by x, the similarities between their mutual elements is denoted by three values:

$$\langle x_{subject}, x_{predicate}, x_{object} \rangle. \tag{8}$$

The similarity values are retrieved through **vector similarity**, as will be discussed in the next Section. The different element weights are denoted as:

$$\{w_{subj}, w_{pred}, w_{obj}\}. \tag{9}$$

Ultimately, the final weight w of the triple t is calculated as follows:

$$w_X = \frac{\sum w_i x_i}{\sum w_i} = \frac{w_{subj}x_{subj} + w_{pred}x_{pred} + w_{obj}x_{obj}}{w_{subj} + w_{pred} + w_{obj}} \tag{10}$$

The weights are initially set as follows: $w_{pred} = 3, w_{subj} = 2, w_{obj} = 1$.

As introduced, the probability of finding a similar triple of type *seen* that can be used to generate a *lexicalisation* of an *unseen* triple is related to the size of the training dataset. It may therefore happen that a suitable replacement cannot be found. In order to clarify this scenario as well, let us consider this example taken from the WebNLG dataset. The system is trying to lexicalise the following *unseen* triple:

$$\langle \text{Aston Martin V8} \mid \text{bodyStyle} \mid \text{Convertible} \rangle \tag{11}$$

A correct output for (11) would be:

$$\text{The Aston Martin V8 is a convertible car.} \tag{12}$$

Within the training dataset, the closest triple to (11) is:

$$\langle \text{Bacon sandwich} \mid \text{dishVariation} \mid \text{BLT} \rangle \tag{13}$$

Therefore, the system would know how to lexicalise (13). Specifically, it would lexicalise in this way:

[11]WebNLG Challenge 2017 Data: https://github.com/ThiagoCF05/webnlg/tree/master/data/v1.5/en

Bacon sandwich is a variation of BLT. (14)

Finally, the original triple elements would be put back, resulting in the final sequence:

Aston Martin V8 is a variation of convertible. (15)

Even if not being completely nonsensical, the final output (12) is far from being a *correct* output; therefore, it is not true that the *Aston Martin V8* is a variation of "convertible", in a literal sense. As we will discuss in the following Section, the similarity between the triples (8) and (11) is too weak; therefore, the approach mentioned above, used for the triple (3), will not be beneficial. In order to face this scenario as well, we take advantage of the "general knowledge" that transfer transformers typically have since they are pre-trained on large amounts of data. It is possible to fine-tune T5 with the WebNLG training dataset. This way, the model can take a RDF triple as an input and produce a suitable *lexicalisation* as an output. Specifically, we are using a submodule named "T5 for Conditional Generation"[12], which enables us to control the text generation and not just activate it, like GPT-based models. The similarity threshold has been determined empirically, looking for the best compromise between the BLEU score maximisation, the human evaluation of generated texts and the number of substitutions performed. Finally, the threshold has been set to 0.7, a value that lowers the Multi-Bleu just slightly (-0.59 BLEU points, -2.23% if compared with DualEnc), letting the model perform a considerable number of substitutions (372 in total) but keeping the human evaluation at a "good" level. Those triples that do not have a proper substitute within the training set will be passed to the T5-based module.

A. Detailed description of the new approach

Our novel proposal is based on what we have already seen during the training stage: given an *unseen* triple, we look for the most similar triple in the training set; then, we use that triple to produce a sentence template, and finally, we put back the surface forms of the original triple. Our pipeline has multiple modules, each addressing a different NLG task [6]. The module that takes care of *unseen* triples will find the most similar *seen* triple, assigning a similarity score to each of the RDF elements (subject, predicate, object). The similarity scores are retrieved through the use of embeddings (vector spaces that capture the semantics of the input by placing semantically similar inputs closer in the space), using the most recent approaches and tools such as RDF2Vec [10] and Gensim[13]. If the module cannot find a proper similar triple, we will take advantage of the "general knowledge" that transfer transformers such as T5 typically have, since they are trained on large amounts of data [21].

Specifically, our architecture is composed of three modules (Fig. 2): (i) a modified version of DualEnc [9]; (ii) a novel similarity module; (iii) T5 for Conditional Generation [8].

DualEnc is responsible for the lexicalisation of all *seen* triples since it reaches state-of-the-art performances in the seen domain. A subtle modification has been made to the original DualEnc model, consisting of a *double triple replacement*: the *unseen* triples are sent to the similarity module, which is responsible for finding the proper replacements; then, a sentence template is calculated by DualEnc, and the original triple is put back before the final *text realisation*. The similarity module comprises several submodules responsible for finding the most similar triple, firstly comparing the predicates, comparing the entities and retrieving similar concepts, and finally assigning a similarity score to each triple. The similarities are calculated with Gensim, specifically with the "keyed vectors" module[14], which provides a method named *most_similar_to_given*(key1, keys_list) that can be used with the keyed vectors version of the DBPedia RDF2Vec [10] model. RDF2Vec [10] is a tool for creating *vector representations of RDF graphs*, which can be utilised in many application scenarios: for instance, it is capable of putting similar entities closer in the vector space than dissimilar ones. Our system uses this strategy to retrieve similar entities and concepts.

Concerning the example in Section V, our system may detect a high similarity between the triples (3) and (5) for the following reasons: the concept of a dog is similar to the concept of a human since they are both mammals, the predicates are semantically close, and there are some shared predicates between the Moon and the ground. Then, we can use the *seen* triple to calculate a template: AGENT-1 | (past tense, third singular person) has stepped on the | PATIENT-1. Finally, we use the template to lexicalise the original triple, obtaining: "Fuffy has stepped on the ground".

If the system is not confident enough about the substitution, the triple is passed to the third module based on T5. The threshold of similarity confidence can be adjusted at will to maximise the BLEU score, the human evaluation or the number of substitutions (in our experiments, we have found a good compromise setting it at 70%, as will be discussed in the next Section). The module based on T5 is pre-trained on the C4[15] and fine-tuned on the WebNLG dataset[16]; it takes a triple as input and produces a fluent *lexicalisation* as an output, without any intermediate step. The usage of a transformer such as T5 is limited in order to minimise

[12]https://huggingface.co/transformers/model_doc/t5.html\#t5forconditionalgeneration

[13]https://github.com/RaRe-Technologies/gensim

[14]https://radimrehurek.com/gensim/models/keyedvectors.html

[15]"Colossal Clean Crawled Corpus", a dataset consisting of hundreds of gigabytes of clean English text scraped from the Web [8] dataset, available for download at: https://huggingface.co/datasets/c4

[16]https://gitlab.com/shimorina/webnlg-dataset/-/tree/master/release_v3.0

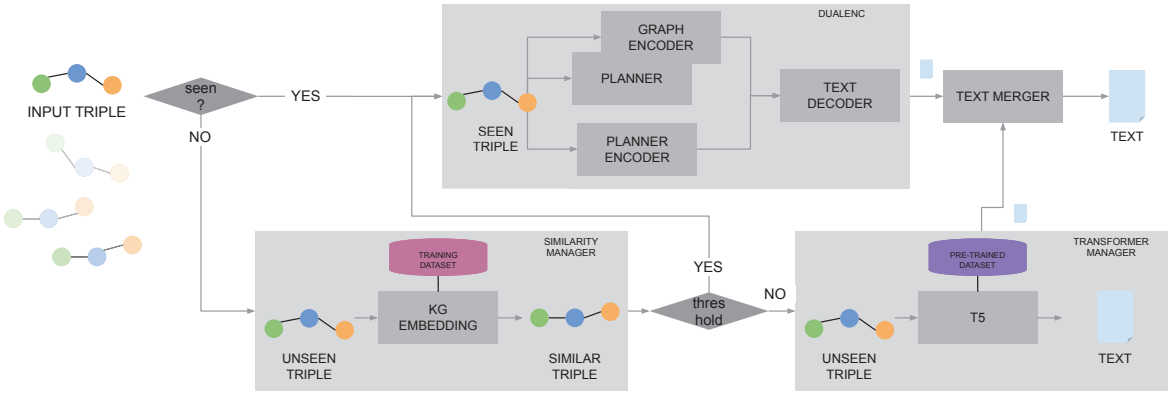

Figure 2. The components of the proposed approach.

the chance of getting hallucinations, omissions and other undesirable effects.

B. Implementation

The novel system is a fork of DualEnc, therefore our additional modules are totally integrated into it and are written in Python. The code is available through a Git repository[17] hosted on BitBucket. In order to run our system, just move to the root directory and run *docker-compose up*. This command will start the training of the original neural planner of DualEnc, then it will run our modified pipeline. Our Gensim-based Flask web server is responsible for finding similarity scores between the triples; it is loaded with a RDF2Vec [10] embedding of the complete DBPedia dump in keyed vectors format. The size of this file is around 29GB, for this reason, it is not shared in the repository. However, it is possible to run our code and get the similarity scores from our self-hosted API.

VI. Evaluation

We have conducted both automatic and human evaluations, as we have found out that automatic metrics alone are not reliable: e.g., many systems get high BLEU scores because "they say wrong things in a good way". This also get worsened by the introduction of T5, which may use different words, such as synonyms, to express the same ideas. For these reasons, we use automatic scores for development purposes but finally, we rely on a human evaluation only. Also, our system will be submitted to the GEM Benchmark, a recently proposed set of tests that also includes a human evaluation.

The **similarity threshold**, which determines whether or not the system relies on a pre-trained model (T5) for *unseen* triple lexicalisation, has been set to **0.7**. This value lowers the Multi-Bleu just slightly (-0.59 BLEU points, -2.23% if compared with DualEnc [9]), letting the model perform a considerable number of substitutions (372 in total) but **keeping the human evaluation at a "good" level** (the human evaluation has been conducted by a group of 20 volunteers, which were asked to classify as good or bad a sample of 60 random-chosen lexicalised *unseen* triples, 10 for each similarity threshold).

A. Automatic evaluation

Automatic evaluation has been done using the official script provided by WebNLG[18], and with SacreBLEU[19] as a double-check. We have conducted two experiments to measure our approach's effectiveness, with and without the pre-trained language-based substitutions module. We choose the same metrics as in the official evaluation of the WebNLG [4] challenge in order to produce comparable results. The results are consistent and are reported in Table II, including the original unmodified DualEnc [9], TripleEnc with pre-trained model deactivated, and finally, the complete TripleEnc with T5 activated.

As expected, the **automatic metrics confirm our hypothesis of a correlation between the *unseen* triples and their corresponding *seen* counterpart**, since the BLEU score is lowered by a small percentage (-6,79% TripleEnc, -11,79% TripleEnc with T5 module), but the number of substitutions is relatively high (372 substitutions made with the similarity module, 928 substitutions made with the T5 module, for a total of 1300 substitutions which is the 48% of the test set). Other automatic metrics are reported in order to make our results comparable with the official WebNLG [4] automatic evaluation.

B. Human evaluation

The human evaluation of NLG systems is an open problem in the state-of-the-art, as discussed in the introduction of this Section. Given the lack of a standard human evaluation metric, we have conducted an evaluation to see whether humans recognise substituted triples as bad lexicalisations or

[17]https://bitbucket.org/ssaleri/tripleenc/

[18]https://github.com/WebNLG/GenerationEval

[19]https://github.com/mjpost/sacrebleu

Table I
MEASURES OF AUTOMATIC AND HUMAN EVALUATIONS, ON DIFFERENT SIMILARITY THRESHOLDS.

Thresh.	BLEU-1	BLEU-2	BLEU-3	BLEU-4	Multi-BLEU	Human Eval.	Subst.s
0.5	48.2	26.9	18.7	11.6	21.74	Bad	1300
0.6	55.4	29.7	19.4	12.5	23.12	Bad	739
0.7	**58.7**	**31.1**	**19.7**	**13.6**	**25.93**	**Good**	**372**
0.8	58.8	31.4	19.7	13.5	26.27	Good	187
0.9	58.8	31.6	19.7	13.5	26.51	Bad	24
1	58.8	31.6	19.7	13.5	26.52	Bad	0

Table II
AUTOMATIC EVALUATION RESULTS ON DIFFERENT METRICS (BLEU, METEOR AND TER).

	BLEU			METEOR			TER		
	Seen	Unseen	All	Seen	Unseen	All	Seen	Unseen	All
DualEnc	63.45	36.73	51.42	0.46	0.37	0.41	0.34	0.55	0.44
TripleEnc	63.45	31.10	47.93	0.46	0.34	0.39	0.34	0.61	0.49
TripleEnc (T5)	63.45	25.76	45.36	0.46	0.27	0.31	0.34	0.72	0.54

accept the output as good. Therefore, to further confirm our hypothesis, we asked a group of 20 volunteers to lexicalise some triples. The group was heterogeneous: some people were already proficient with this kind of problem, some people have never dealt with NLG before. The only requirement was proficiency in the English language. The experiments were conducted in reverse order to better introduce the problem to our volunteers (this also give us the possibility of filtering out the answers from the people who did not understand the problem). Results have shown how even a simple triple is lexicalised in different ways by different humans.

VII. CONCLUSIONS

Despite being fascinating, Natural Language Processing (NLP) and NLG raise broad problems that still need to be investigated. We believe that machines will be able to talk like humans very soon; natural language technologies are already available and have succeeded in making people embrace modern technologies seamlessly (e.g., technologies like Amazon Alexa and Google Home), but there is still a long way to go. In this work, we have reviewed systems from the past, starting from the early days of NLG, ending up with the most innovative technologies. We discussed **pipelines** and recent **end-to-end** systems, we investigated **sequential, graph and pre-trained** approaches, experiencing the problems they were trying to solve. Multiple problems were discussed along with their proposed solutions with particular attention to the drop of performances on the **unseen domain** and the **hallucinations**. These problems make the entire system unreliable, thus unusable in a real-world scenario. We targeted these problems, and finally, we proposed our solution. We have made a hypothesis regarding similarities between triples, and we have measured to which extent it is verified. TripleEnc was designed specifically to overcome the difficulties of modern approaches; it is based on the state-of-the-art model known as DualEnc and is proposed as its improved successor. Our approach does not stick with a particular model; therefore, it can be used with any template-based model. Our experiments confirmed the validity of our hypothesis, showing a correlation between similar RDF-triples and their corresponding text realisations, as we were expecting. The results showed minor lexicalisations improvements compared with the current state-of-the-art model.

Our model and our experiments will be used as a basis for the development of a future model. TripleEnc will be further investigated and improved. Therefore it will be submitted to the next WebNLG [24] challenge. Several improvements strategies could be investigated, such as the usage of a language model for **syntax checking** and **syntax correcting**. A further improvement could be given by the usage of a discriminator, able to measure the quality of generated sentences, as we have seen with Generative Adversarial Network (GAN)s. Ultimately, an additional improvement could be given by adding a strategy for treating *unseen* triples as *seen*, once the system learns how to lexicalise them, in a similar approach used with the Reinforcement Learning paradigm.

REFERENCES

[1] A. Gatt and E. Krahmer, "Survey of the state of the art in natural language generation: Core tasks, applications and evaluation," *J. Artif. Int. Res.*, vol. 61, no. 1, pp. 65–170, Jan. 2018.

[2] A. Graefe, "Guide to automated journalism," Tow Center for Digital Journalism, Columbia University, Tech. Rep., 2016.

[3] K. N. Dörr, "Mapping the field of algorithmic journalism," *Digital Journalism*, vol. 4, no. 6, pp. 700–722, 2016.

[4] C. Gardent, A. Shimorina, S. Narayan, and L. Perez-Beltrachini, "The webnlg challenge: Generating text from rdf data," in *Proceedings of the 10th International Conference on Natural Language Generation*. Association for Computational Linguistics, 2017, pp. 124–133.

[5] B. D. Trisedya, J. Qi, R. Zhang, and W. Wang, "GTR-LSTM: A triple encoder for sentence generation from RDF data," in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Melbourne, Australia: Association for Computational Linguistics, Jul. 2018, pp. 1627–1637.

[6] E. Reiter and R. Dale, "Building applied natural language generation systems," *Natural Language Engineering*, vol. 3, no. 1, p. 57–87, 03 2002.

[7] T. Castro Ferreira, C. van der Lee, E. van Miltenburg, and E. Krahmer, "Neural data-to-text generation: A comparison between pipeline and end-to-end architectures," in *Proceedings of the EMNLP-IJCNLP*. Hong Kong, China: Association for Computational Linguistics, Nov. 2019, pp. 552–562.

[8] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu, "Exploring the limits of transfer learning with a unified text-to-text transformer," *CoRR*, 2019.

[9] C. Zhao, M. Walker, and S. Chaturvedi, "Bridging the structural gap between encoding and decoding for data-to-text generation," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Online: Association for Computational Linguistics, Jul. 2020, pp. 2481–2491.

[10] P. Ristoski, J. Rosati, T. Di Noia, R. De Leone, and H. Paulheim, "Rdf2vec: Rdf graph embeddings and their applications," *Semantic Web Journal*, 2018.

[11] H. Gao, L. Wu, P. Hu, and F. Xu, "Rdf-to-text generation with graph-augmented structural neural encoders," in *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20*, C. Bessiere, Ed. International Joint Conferences on Artificial Intelligence Organization, 7 2020, pp. 3030–3036, main track.

[12] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu, "Bleu: a method for automatic evaluation of machine translation," in *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*. Philadelphia, Pennsylvania, USA: Association for Computational Linguistics, Jul. 2002, pp. 311–318.

[13] E. Reiter and A. Belz, "An investigation into the validity of some metrics for automatically evaluating natural language generation systems," *Computational Linguistics*, vol. 35, no. 4, pp. 529–558, 2009.

[14] S. Banerjee and A. Lavie, "METEOR: An automatic metric for MT evaluation with improved correlation with human judgments," in *Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*. Ann Arbor, Michigan: Association for Computational Linguistics, Jun. 2005, pp. 65–72.

[15] M. Snover, B. Dorr, R. Schwartz, L. Micciulla, and J. Makhoul, "A study of translation edit rate with targeted human annotation," in *Proceedings of the 7th Conference of the Association for Machine Translation in the Americas: Technical Papers*. Cambridge, Massachusetts, USA: Association for Machine Translation in the Americas, Aug. 8-12 2006, pp. 223–231.

[16] A. Rohrbach, L. A. Hendricks, K. Burns, T. Darrell, and K. Saenko, "Object hallucination in image captioning," in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Brussels, Belgium: Association for Computational Linguistics, Oct.-Nov. 2018, pp. 4035–4045.

[17] D. Marcheggiani and L. Perez-Beltrachini, "Deep graph convolutional encoders for structured data to text generation," in *Proceedings of the 11th International Conference on Natural Language Generation*. Tilburg University, The Netherlands: Association for Computational Linguistics, Nov. 2018, pp. 1–9.

[18] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," *CoRR*, 2016.

[19] K. Xu, L. Wu, Z. Wang, Y. Feng, and V. Sheinin, "Graph2seq: Graph to sequence learning with attention-based neural networks," *CoRR*, 2018.

[20] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," *CoRR*, 2015.

[21] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," *CoRR*, 2017.

[22] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. M. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei, "Language models are few-shot learners," *CoRR*, 2020.

[23] M. Imai, D. Gentner, and N. Uchida, "Children's theories of word meaning: The role of shape similarity in early acquisition," *Cognitive Development*, vol. 9, no. 1, pp. 45–75, 1994.

[24] T. Castro Ferreira, C. Gardent, N. Ilinykh, C. van der Lee, S. Mille, D. Moussallem, and A. Shimorina, "The 2020 bilingual, bi-directional WebNLG+ shared task: Overview and evaluation results (WebNLG+ 2020)," in *Proceedings of the 3rd International Workshop on Natural Language Generation from the Semantic Web (WebNLG+)*. Dublin, Ireland (Virtual): Association for Computational Linguistics, 12 2020, pp. 55–76.