The Natural Language Generation Pipeline, Neural Text Generation and Explainability

Juliette Faille, Albert Gatt, Claire Gardent

The Natural Language Generation Pipeline, Neural Text Generation and Explainability

Juliette Faille
CNRS/LORIA
Nancy, France
juliette.faille@loria.fr

Albert Gatt
University of Malta
Msida, Malta
albert.gatt@um.edu.mt

Claire Gardent
CNRS/LORIA
Nancy, France
claire.gardent@loria.fr

Abstract

End-to-end encoder-decoder approaches to data-to-text generation are often black boxes whose predictions are difficult to explain. Breaking up the end-to-end model into sub-modules is a natural way to address this problem. The traditional pre-neural Natural Language Generation (NLG) pipeline provides a framework for breaking up the end-to-end encoder-decoder. We survey recent papers that integrate traditional NLG sub-modules in neural approaches and analyse their explainability. Our survey is a first step towards building explainable neural NLG models.

1 Motivation

The end-to-end encoder-decoder is a popular neural approach that is efficient to generate fluent texts. However it has often been shown to face some adequacy problems such as hallucination, repetition or omission of information. As the end-to-end encoder-decoder approaches are often "black box" approaches, such adequacy problems are difficult to understand and solve.

In contrast, pre-neural NLG has often integrated a number of sub-modules implementing three main NLG sub-tasks (Reiter and Dale, 2000): macroplanning ("What to say"), microplanning and surface realisation ("How to say").

To improve adequacy and provide for more explainable approaches, recent work has proposed integrating traditional pre-neural NLG sub-modules into neural NLG models. In this paper, we survey some[1] of this work, focusing mainly on generation from data- and meaning representations[2]. Table 1 lists the approaches we consider. We start by identifying which NLG sub-tasks have been modeled in these approaches using which methods (Sec. 2-4). We then go (Sec. 5) on to briefly discuss to which extent the methods used by each of these models may facilitate explainability.

2 Macroplanning

Macroplanning is the first subtask of the traditional pre-neural NLG pipeline. It answers the "what to say" question and can be decomposed into selecting and organising the content that should be expressed in the generated text.

2.1 Content Determination

Content determination is the task of selecting information in the input data that should be expressed in the output text. The importance of this subtask depends on the goal of a generation model. In the papers surveyed, papers which verbalise RDF or Meaning Representations (MR) input do not perform content determination, while Shen et al. (2019), who generate headlines from source text, do.

In this approach, content selection is viewed as a sequence labelling task where masking binary latent variables are applied to the input. Texts are generated by first sampling from the input to decide which content to cover, then decoding by conditioning on the selected content. The proposed content selector has a ratio of selected tokens that can be adjusted, bringing controllability in the content selection.

It should also be noted that in template-based approaches such as (Wiseman et al., 2018), which use templates for text structuring (cf. Sec. 2.2), the template choice determines the structure of the output text but also has an influence on the content selection since some templates will not express some of the input information. For instance, the output 2 in

[1] Given the space limitations, the survey is clearly not exhaustive.

[2] We also include (Shen et al., 2019)'s model for text-to-text generation as it provides an interesting module for content selection which few of the papers we selected address.

Contribution	Content Selection	Document Structuring	REG	Input
Moryossev 19b		**Supervised**		RDF triples
Moryossev 19a		**Supervised**	**Rule-based with LM score**	RDF triples
Sha 17		**Attention**		Tables
Ferreira 19		**Supervised**	**Neural**	RDF triples
Laha 20			Rule-based	RDF triples
Gehrmann 18		**LV Template**	Coverage and length penalty	MR
Shen 20		**LV Hierarchical + Attention**		RDF triples
Shen 19	**LV**			Text
Wiseman 18	LV Template	**LV Template**		Tables
Zhao 20		**Supervised**		RDF triples
Shao 19		**LV Hierarchical**	Plan variations	Tables
Distiawan 18		**Structure encoding**		RDF triples

Table 1: Summary of the NLG models for the sub-tasks Content Selection, Document structuring and REG. The bold types indicates the main sub-task(s) modeled in each contribution and normal type the sub-task(s) that are of lesser importance in the contribution. The input type is given in the last column. LV stands for Latent Variable.

Table 2 does not include the input customer rating information.

2.2 Document structuring

Document structuring is the NLG sub-task in which the previously selected content is ordered and divided into sentences and paragraphs. The goal of this task is to produce a text plan. Many approaches choose to model document structuring. Four main types of approaches can be distinguished depending on whether the content plan is determined by latent variables, explicit content structuring, based on the input structure or guided by a dedicated attention mechanism.

Latent Variable Approaches One possible way to model content structure is to use latent variables.

Wiseman et al. (2018) introduce a novel, neural parameterization of a hidden semi-markov model (HSMM) which models latent segmentations in an output sequence and jointly learns to generate. These latent segmentations can be viewed as templates where a template is a sequence of latent variables (transitions) learned by the model on the training data. Decoding (emissions) is then conditioned on both the input and the template latent variables. Intuitively, the approach learns an alignment between input tokens, latent variables and output text segments (cf. Table 2). A key feature of this approach is that this learned alignment can be used both to control (by generating from different templates) and to explain (by examining the mapping between input data and output text mediated by the latent variable) the generation model.

Similarly, Gehrmann et al. (2018) develop a mixture of models where each model learns a latent sentence template style based on a subset of the input. During generation and for each input, a weight is assigned to each model. For the same input information, two templates could produce the outputs "There is an expensive British restaurant called the Eagle" and "The Eagle is an expensive British Restaurant". The template selection defines in which order the information should be expressed and therefore acts as a plan selection.

Latent variable approaches have also been proposed for so-called hierarchical approaches where the generation of text segments, generally sentences, is conditioned on a text plan. Thus, Shen et al. (2020) propose a model where, given a set of input records, the model first selects a data record based on a transition probability which takes into account previously selected data records and second, generates tokens based on the word generation probability and attending only to the selected data record. This "strong attention" mechanism allows control of the output structure. It also reduces hallucination by using the constraints that all data records must be used only once. The model automatically learns the optimal content planning by exploring exponentially many segmentation/correspondence possibilities using the forward algorithm and is end-to-end trainable.

Similarly Shao et al. (2019) decompose text generation into a sequence of sentence generation subtasks where a planning latent variable is learned based on the encoded input data. Using this latent variable, the generation is made hierarchically with a sentence decoder and a word decoder. The plan decoder specifies the content of each output sentence. The sentence decoder also improves high-level planning of the text. Indeed this model helps capture inter-sentence dependencies in particular

Input	name[Travellers Rest Beefeater], customerRating[3 out of 5], area[riverside], near[Raja Indian Cuisine].
Output 1	[Travellers Rest Beefeater]$_{55}$ [is a]$_{59}$ [3 star]$_{43}$ [restaurant]$_{11}$ [located near]$_{25}$ [Raja Indian Cuisine]$_{40}$ [.]$_{53}$
Template 1	$z^i = \langle 55, 59, 43, 11, 25, 40, 53 \rangle$.
Output 2	[Travellers Rest Beefeater]$_{55}$ [is a]$_{59}$ place to eat]$_{12}$ [located near]$_{25}$ [Raja Indian Cuisine]$_{40}$ [.]$_{53}$
Template 2	$z^i = \langle 55, 59, 12, 25, 40, 53 \rangle$.

Table 2: Example templates and outputs segmentation from (Wiseman et al., 2018)'s approach

thanks to the global planning latent variable and attention mechanisms in the sentence decoder.

Remark. Learning a template can cover different NLG subtasks at once. For instance Gehrmann et al. (2018) use sentence templates, which determine the order in which the selected content is expressed (document structuring), define aggregation and for some cases encourage the use of referring expressions and of some turns of phrase (usually included in the lexicalisation sub-task) and defines to some extent the surface realization.

Explicit Content Structuring using Supervised Learning. Other approaches explicitly generate content plans using supervised learning.

In (Moryossef et al., 2019b), a text plan is a sequence of sentence plans where each sentence plan is an ordered tree. Linearisation is then given by a pre-order traversal of the sentence trees. The authors adopt an overgenerate-and-rank approach where the text plans are generated using symbolic methods and ranked using a product of expert model integrating different probabilities such as the relation direction probability (e.g. the probability that the triple {A, manager, B} is expressed as "A is the manager of B" or, in reverse order, as "B is managed by A") or the relation transition probability (which relations are usually expressed one after the other, e.g. birth place and birth date). Moryossef et al. (2019a) propose a variant of this model where the generation and choice of the plan to be realized is done by a neural network controller which uses random truncated DFS traversals. This new planner is achieving faster performance compared to (Moryossef et al., 2019b).

In (Castro Ferreira et al., 2019) templates are lists of ordered triples divided into sentences. Castro Ferreira et al. (2019) first order the input triples in the way they will be expressed and then divides this ordered list into sentences and paragraphs. This ordering of triples and segmentation into sentences is studied with different models : two rule-based baselines (which apply either random selection of triples or most frequent order seen on the training set) and two neural models (GRU and Transformer). They show that neural models perform better on the seen data but do not generalize well on unseen data.

Zhao et al. (2020) model a plan as a sequence of RDF properties which, before decoding, is enriched with its input subject and object. A Graph Convolutional Network (GCN) encodes the graph input and a Feed Forward Network is used to predict a plan which is then encoded by an LSTM. The LSTM decoder takes as input the hidden states from both encoders. In this approach the document structuring sub-task is tackled by an additional plan encoder.

Input structure encoding Some approaches use the structure of the input to constrain the order in which input units are verbalised. Thus, Distiawan et al. (2018) capture the inter and intra RDF triples relationships using a graph-based encoder (GRT-LSTM). It then combines topological sort and breadth-first traversal algorithms to determine in which order the vertices of the GRT-LSTM will be input with data during training thereby performing content planning.

Dedicated Attention mechanisms Instead of encoding input structure, some of the approaches use attention mechanisms to make their model focus on specific aspects of the data structure. Sha et al. (2018) take advantage of the information given by table field names and by relations between table fields. They use a dispatcher before the decoder. The dispatcher is a self-adaptative gate that combines content-based attention (on the content of the field and on the field name of the input table) and link-based attention (on the relationships between input table fields).

3 Microplanning

Microplanning is the NLG sub-task which aims at defining "how to say" the information that was selected and structured during macroplanning.

3.1 Referring Expression Generation (REG)

Few approaches explicitely model the REG subtasks. In (Moryossef et al., 2019a), REG is handled in a postprocessing step, using names for first mentions, and subsequently the pronoun or string with the highest BERT LM score. Similarly, Laha et al. (2020) use heuristic sentence compounding and coreference replacement modules as postprocessing steps. Castro Ferreira et al. (2019) explore both a the baseline model which systematically replaces delexicalised entities with their Wikipedia identifiers and the integration in the NLG pipeline of the NeuralREG model (Castro Ferreira et al., 2018). NeuralREG uses two bidirectional LSTM encoders which encode the pre- and post-contexts of the entity to be referred to. An LSTM decoder with attention mechanisms on the pre- and post-contexts generates the referring expression. Gehrmann et al. (2018) use copy-attention to fill in latent slots inside of learned templates where slots are most to be filled with named entities.

3.2 Lexicalisation

Lexicalisation maps input symbols to words. In neural approach, lexicalisation is mostly driven by the decoder which produces a distribution over the next word, from which a lexical choice is made. The copy mechanism introduced by See et al. (2017) is also widely used as it allows copying from the input (Sha et al., 2018; Moryossef et al., 2019b; Laha et al., 2020). At each decoding step, a learned "switch variable" is computed to decide whether the next word should be generated by the S2S model or simply copied from the input. Inspecting the value of the switch variable permits assessing how much lexicalisation tends to copy vs to generate and can provide some explainability in the lexicalisation sub-task. Finally, a few approaches use lexicons and rule-based mapping. In particular, Castro Ferreira et al. (2019) use a rule-based model to generate the verbalization of RDF properties.

4 Surface realisation

Surface realisation is the last NLG task and consists in creating a syntactically well-formed text out of the representations produced by the previous step. While surface realisation is at the heart of generation when generating from meaning representations, it is largely uncharted in data- and table-to-text NLG and results either from the decoder language model (which decides on the words and thereby indirectly on the syntax of the generated text) or from the templates used for generation (Castro Ferreira et al., 2019; Moryossef et al., 2019b; Wiseman et al., 2018).

5 Conclusion

Explainable models enable a clear understanding of how the output generated by the model relates to its input. In this short paper, we surveyed a number of neural data-to-text generation models which implement some or all of the NLG pipeline sub-tasks with the aim of identifying methods which could help enhance explainability in neural NLG.

Our survey highlights two main ways of enhancing explainability: explicit intermediate structures produced by neural modules modeling the NLG pipeline subtasks or latent variables modeling the interface between these modules.

Thus (Castro Ferreira et al., 2019)'s supervised pipeline model outputs content plans, sentence templates and referring expressions which can all be examined, quantified and analysed thereby supporting a detailed qualitative analysis of each subtasks. Similarly, Moryossef et al. (2019b,a) output explicit text plans and text plan linearisations and Zhao et al. (2020) text plans.

In contrast, the models introduced in (Shao et al., 2019; Wiseman et al., 2018; Gehrmann et al., 2018; Shen et al., 2019, 2020) are based on latent variables which mediate the relation between input and output tokens and intuitively, model a document plan by mapping e.g., input RDF triples to text fragments. As illustrated in Table 2 which shows examples of latent templates used to generate from the input, latent variables provide a natural means to explain the model's behaviour i.e., to understand which part of the input licenses which part of the output. They are also domain agnostic and, in contrast to the explicit pipeline models mentioned in the previous paragraph, they do not require the additional creation of labelled data which often relies on complex, domain specific, heuristics.

A third alternative way to support explainability is model analysis such as supported e.g., by the AllenNLP Interpret toolkit (Wallace et al., 2019) which provides two alternative means for interpreting neural models. Gradient-based methods explain a model's prediction by identifying the importance of input tokens based on the gradient of the loss with respect to the tokens (Simonyan et al., 2014)

while adversarial attacks highlight a model's capabilities by selectively modifying the input.

In future work, we plan to investigate whether domain agnostic, linguistically inspired intermediate structures such as meaning representations could be used to both support explainability and improve performance. Another interesting direction for further research would be to develop common evaluation benchmarks and metrics to enable a detailed analysis and interpretation of how neural NLG models perform for each of the NLG pipeline sub-tasks. Finally, while most of the approaches we surveyed concentrate on modeling the interaction between content planning and micro-planning, it would be useful to investigate whether any of the methods highlighted in this paper could be exploited to explore and improve the explainability of the various micro-planning sub-tasks (lexicalisation, aggregation, regular expression generation, surface realisation).

Acknowledgments

We would like to thank the anonymous reviewers for their helpful comments. Research reported in this publication is part of the project NL4XAI. This project has received funding from the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement No 860621. This document reflects the views of the author(s) and does not necessarily reflect the views or policy of the European Commission. The REA cannot be held responsible for any use that may be made of the information this document contains.

References

Thiago Castro Ferreira, Chris van der Lee, Emiel van Miltenburg, and Emiel Krahmer. 2019. Neural data-to-text generation: A comparison between pipeline and end-to-end architectures. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 552–562, Hong Kong, China. Association for Computational Linguistics.

Thiago Castro Ferreira, Diego Moussallem, Ákos Kádár, Sander Wubben, and Emiel Krahmer. 2018. NeuralREG: An end-to-end approach to referring expression generation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1959–1969, Melbourne, Australia. Association for Computational Linguistics.

Bayu Distiawan, Jianzhong Qi, Rui Zhang, and Wei Wang. 2018. 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)*, pages 1627–1637.

Sebastian Gehrmann, Falcon Dai, Henry Elder, and Alexander Rush. 2018. End-to-end content and plan selection for data-to-text generation. In *Proceedings of the 11th International Conference on Natural Language Generation*, pages 46–56, Tilburg University, The Netherlands. Association for Computational Linguistics.

Anirban Laha, Parag Jain, Abhijit Mishra, and Karthik Sankaranarayanan. 2020. Scalable micro-planned generation of discourse from structured data. *Computational Linguistics*, 45(4):737–763.

Amit Moryossef, Yoav Goldberg, and Ido Dagan. 2019a. Improving quality and efficiency in plan-based neural data-to-text generation. In *Proceedings of the 12th International Conference on Natural Language Generation*, pages 377–382, Tokyo, Japan. Association for Computational Linguistics.

Amit Moryossef, Yoav Goldberg, and Ido Dagan. 2019b. Step-by-step: Separating planning from realization in neural data-to-text generation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2267–2277, Minneapolis, Minnesota. Association for Computational Linguistics.

Ehud Reiter and Robert Dale. 2000. *Building natural language generation systems*. Cambridge university press.

Abigail See, Peter J. Liu, and Christopher D. Manning. 2017. Get to the point: Summarization with pointer-generator networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1073–1083, Vancouver, Canada. Association for Computational Linguistics.

Lei Sha, Lili Mou, Tianyu Liu, Pascal Poupart, Sujian Li, Baobao Chang, and Zhifang Sui. 2018. Order-planning neural text generation from structured data. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, pages 5414–5421. AAAI Press.

Zhihong Shao, Minlie Huang, Jiangtao Wen, Wenfei Xu, and Xiaoyan Zhu. 2019. Long and diverse text

generation with planning-based hierarchical variational model. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3257–3268, Hong Kong, China. Association for Computational Linguistics.

Xiaoyu Shen, Ernie Chang, Hui Su, Cheng Niu, and Dietrich Klakow. 2020. Neural data-to-text generation via jointly learning the segmentation and correspondence. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7155–7165, Online. Association for Computational Linguistics.

Xiaoyu Shen, Jun Suzuki, Kentaro Inui, Hui Su, Dietrich Klakow, and Satoshi Sekine. 2019. Select and attend: Towards controllable content selection in text generation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 579–590, Hong Kong, China. Association for Computational Linguistics.

Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. 2014. Deep inside convolutional networks: Visualising image classification models and saliency maps. In *ICLR*, Banff, Canada.

Eric Wallace, Jens Tuyls, Junlin Wang, Sanjay Subramanian, Matt Gardner, and Sameer Singh. 2019. AllenNLP interpret: A framework for explaining predictions of NLP models. In *EMNLP*, pages 7–12, Hong Kong, China.

Sam Wiseman, Stuart Shieber, and Alexander Rush. 2018. Learning neural templates for text generation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3174–3187, Brussels, Belgium. Association for Computational Linguistics.

Chao Zhao, Marilyn Walker, and Snigdha Chaturvedi. 2020. 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*, volume 1.