Neural Micro-Planning for Data to Text Generation Produces more Cohesive Text

Roy Eisentadt, Michael Elhadad
Dept. of Computer Science, Ben-Gurion University of the Negev
Beer Sheva, Israel
royeis@post.bgu.ac.il, elhadad@cs.bgu.ac.il

Abstract

We aim to prove the usefulness of separating data to text generation into micro-planning and realization, and focus on micro-planning as a task that can be learned and evaluated separately. We adopt a simple structure for micro-plans and develop an initial neural model to learn such a plan from a flat input of triplets. We define a method to measure planning quality, and an evaluation method of generated text by examining syntactic phenomena related to text cohesion. In experiments on the WebNLG dataset, we demonstrate the correlation between higher quality planning and more natural, cohesive text. The quantitative data-driven methodological approach we illustrate can help formulate hypotheses that a more sophisticated micro-plan formalism and its interface with surface realization decisions could help explore.

1 Introduction

Traditional NLG pipelines distinguish distinct sub-tasks addressed by a generation system, including content determination, text structuring, sentence aggregation, lexicalization and surface realization (Gatt and Krahmer, 2018). Recent work on neural NLG has blurred the distinction among these sub-tasks and encouraged data-driven end-to-end approaches, such as transformer-based encoder-decoder architectures (Lewis et al., 2020). Recent data to text generation approaches are revisiting this decision, and show the benefit of dividing the full task into two steps: planning and realization (Moryossef et al., 2019; Castro Ferreira et al., 2019). The goal of micro-planning is to organize the input raw data into an interpretable and coherent information structure. Realization is then applied on this structure to generate coherent text that covers all the expected content without redundancy and without introducing unintended content. Planning and realization deal with distinct but closely related aspects of text structuring: planning is related to concepts from discourse theory such as rhetorical structure, information flow and coherence while realization handles the lexical and syntactic aspects of these concepts including information packaging, clause structuring and aggregation.

A modular approach brings two benefits: more control over each component of the generation and simpler modeling of both sub-problems when compared to end to end models. We hypothesize that an explicit planning model, including quality evaluation of plans, can lead to better control of the generated text in data-to-text tasks, and boost performance, as was indeed demonstrated in (Moryossef et al., 2019). We revisit this modularity argument with three new directions: (1) we study the extent to which a robust learned planning module can be derived (as opposed to a rule-based planning method); (2) we investigate whether an independent planning quality metric can be established, and the extent to which it correlates with end to end text quality metrics; (3) finally, we investigate specific aspects in realization that are directly related to micro-planning and cohesion and the extent to which good plans control their usage.

Applying learning methods to solve the task of planning is difficult for two main reasons: (1) available datasets (Gardent et al., 2017a,b) do not reward variability in plans. They contain a few pairs (data, text) for a given input (usually 3 to 5 variants per input), but there is no incentive to demonstrate a variety of plans to realize the same input; an ideal dataset to learn planning would instead hold different paraphrases for each entry based on changes in micro-planning; (2) Given a target text to generate, micro-plannings are not observable. One can come up with methods to derive a plan from a given text but the nature of the plan, how it is related to

observable syntactic structure is essentially an internal decision. Despite these obstacles, we aim to demonstrate the usefulness of learning an intermediate plan representation for data to text generation. Beyond this quantitative architectural analysis, the experimental setting we investigate provides a useful platform to explore more sophisticated models of text planning.

2 Datasets

Our experiments are based on the WebNLG 2020 dataset (Gardent et al., 2017a) which provides knowledge-graph to text entries. Each knowledge graph is composed of a set of logical forms that are encoded as triplets: (sub, rel, obj). For the purpose of learning and evaluation of the planning task, we utilize the DeepNLG dataset (Castro Ferreira et al., 2019), which is based on WebNLG 2017 and associates, for each (data, text) pair a manually derived plan, which has the structure of an ordered sequence of groups of triplets (one per observed sentence in the text). This data allows us to train in a supervised manner on the data-to-plan task.

3 Modeling Plans

We hypothesize that the task of generating text from a set of triplets $\mathcal{T} = t_1, \ldots, t_n$ will be improved if we model it as a pipeline of two stages $\mathcal{T} \rightarrow \mathcal{P}lan$ and $\mathcal{P}lan \rightarrow \mathcal{T}ext$.

Since plans are not observed, we need to decide how to model them. One can distinguish two strategies for this decision: (1) latent transition-based model and (2) representation-based. In the latent approach, we apply a neural encoder-decoder architecture with a transition-based model for planning similar to (Nivre et al., 2004). The neural encoder creates a latent representation of the input $\mathcal{T}$. Conditioned on this representation, the decoder works in a gradual manner to perform predefined actions that correspond to micro-planning decisions. These actions include generating a word, deciding to aggregate two relations, closing a sentence and starting to generate a new one. In this approach, there is no explicit representation of plans, instead, the model maps a latent representation of the input structure to discrete micro-planning decisions. The exact list of these decisions corresponds to the claims of a text planning theory.

In contrast, a representation-based approach separates this procedure into two well defined substeps. In a first step, the input data is mapped to a plan. This plan describes the order between atomic units into sentences, and packaging of the data into a sentence-level micro-plan. In a second step, given this plan natural text is generated. We refer to these two models as planner and realizer.

In the second approach, we must select a formalism to represent plans, which depends both on the input data and on the nature of the desired generated text. Furthermore, this approach requires data that explicitly represents such plans or from which plans can be derived to allow learning in a supervised manner. An obvious advantage of this approach is that both tasks are of lower complexity than end to end data to text which should make them easier to learn. Another advantage is better interpretability. One can measure the efficiency of plans generation and measure the correlation between quality of plans and the success of the overall task.

In this work, we explore the planner-realizer approach with a definition of plans as derived from the DeepNLG dataset. We formalize the task of WebNLG planning as follows: Given a set of triplets $\mathcal{T} = \{t_1, ..., t_m\}$ output an ordered list of ordered lists $p = (s_1, ..., s_n)$ such that $\bigcup_{i=1}^{n} s_i = \mathcal{T}$ and for each $i, j \in \{1, ..., n\}$, such that $i \neq j$, $s_i \cap s_j = \phi$.
The task consists of: (1) grouping triplets into sentences; (2) determining the order of sentences; (3) determining the order of triplets within each sentence. This definition does not provide an explicit measure of plan quality, we start by investigating what would be a good metric to assess plan quality. The choice of this simple plan formalism is an initial operational step, which has the benefit of relying on existing data. In the future, we will explore different representation formalisms for plans, and their connection to the decisions made by the realizer. We expect that document plan theories explaining information flow and packaging will provide fertile ground for this work (Kuppevelt, 1996; Roberts, 2012).

4 Plan Quality Measure

In order to maximize the benefit from the separation of planning from realization, we need to measure the intermediate success in the generation of plans. Given a tool to measure the quality of constructed plans that is known to be correlated with the quality of the output text, one can com-

pare different approaches to planning, and enhance results of a complete data-to-text pipeline.

We propose such a metric that evaluates a candidate plan against a set of reference plans as observed in the data (such as DeepNLG). Our metric combines two aspects: ordering consistency and grouping consistency. Given a plan $p = (s_1, ..., s_n)$ constructed from m triplets, denote $p_{flat} := s_1 \circ s_2, ..., \circ s_n = (t_{i_1}, ...t_{i_m})$, the ordered concatenation of all items in p. p_{flat} corresponds to the ordered list of triplets as would appear in the generated text according to plan p without considering grouping them into sentences. Given a candidate and a reference plan $\hat{p} = (s'_1, ..., s'_n)$ and $p = (s_1, ..., s_n)$ involving m triplets, we denote x_i and y_i as the positions or indices of t_i within the ordered lists $\hat{p}_{flat}$ and p_{flat} respectively for any $1 \leq i \leq m$ (both plans are of the same length – the shorter plan padded with empty lists when needed). We use Kendall's ranking correlation coefficient to define:

$$\tau(\hat{p}, p) = \frac{\sum_{i<j} sign(x_i - x_j) sign(y_i - y_j)}{\binom{m}{2}}.$$

Next we define grouping accuracy:

$$\alpha(\hat{p}, p) = \frac{\sum_{i=1}^{n} s'_i \cap s_i}{m}$$

A higher value of $\tau(\hat{p}, p)$ indicates similarity in the triplets' order of appearance between the candidate and reference plan. A higher value of $\alpha(\hat{p}, p)$ indicates similar grouping of triplets into sentences and similar ordering of sentences.

We combine these two aspects in a convex combination to define the plan quality metric PQM:

$$PQM(\hat{p}, p) = \lambda \cdot \frac{\tau(\hat{p}, p) + 1}{2} + (1 - \lambda) \cdot \alpha(\hat{p}, p)$$

When a candidate plan $\hat{p}$ is measured against a set of reference plans $\mathcal{P} = \{p_1, .., p_k\}$, we define $PQM(\hat{p}, \mathcal{P}) = \max_{1 \leq i \leq k} PQM(\hat{p}, p_i)$.

The value of the parameter λ is empirically chosen to be 0.7. In order to determine this value, we select the value which provides the highest correlation with the end-to-end text quality as measured by the BLEU metric for a realizer that takes a plan as input. To perform these steps, we train two distinct models: (1) a planner trained on the DeepNLG development set entries; (2) a realizer which generates the observed text given a DeepNLG plan as input. We train both models with a T5 text-to-text transformer model similar to (Kale, 2020). Given this pipeline, we computed PQM_λ for λ in $(0.1 \ldots 0.9)$ and the BLEU score per entry. This procedure provides PQM_λ estimations and a single BLEU score per entry in the development set. Pearson's correlation measure for each value of λ between PQM_λ and BLEU scores lets us pick the optimal λ.

model	BLEU	PQM
T5	44.44	–
T5 teacher exposure	47.50	–
T5 planner-realizer	55.01	0.838

Table 1: Model evaluation

	E2E	P+R	refs
#words	38.77	21.67	22.60
#sentences	3.42	1.34	1.45
#coordinated NPs	0.15	0.33	0.28
#coordinated VPs	0.12	0.32	0.21
#relative clauses	0.11	0.29	0.24
#subordinate clauses	0.06	0.16	0.19
#coordinate clauses	0.05	0.12	0.11

Table 2: Syntactic Phenomena Frequencies: E2E - T5 end-to-end, P+R - T5 planner + realizer

5 Experiments

We compare baseline data-to-text models which are trained to map end-to-end WebNLG 2020 data to text using the same T5 transformer-based architecture with the modular architecture (Planner, Realizer) where each of the modules is trained separately. We compare two end-to-end baselines: The first is a pre-trained T5 model which has shown promising results on data-to-text (Kale, 2020). It is fine-tuned to generate text given input triplets. In the second baseline, we use the same T5 backbone with a teacher exposure strategy during fine-tuning: each entry is composed of the input triplets as before concatenated with an incomplete prefix of the desired reference text that contains complete sentences. In this approach, the model learns to complete text given all triplets and a text prefix. The third model is the modular (Planner, Realizer) pipeline described above. Results (Table 1) indicate overall improvement in BLEU scores when using the modular approach.

To assess the impact of better controlling planning, we specifically investigate syntactic aspects of the generated text related to text cohesion (Halliday and Hasan, 1976). Indeed, planning does not determine all aspects of realization - for example, it does not impact the lexicalization of entities encoded in triplets, but it does impact directly sen-

tence packaging, aggregation, coordination, clause structure, relative clauses. Table 2 compares text generated by the baseline end-to-end model, the modular model (Planner, Realizer) and the reference texts of WebNLG 2020. It shows the frequency of different syntactic phenomena per sentence in the text as well as number of words and sentences in each text. To identify occurrences of these phenomena, we used spaCy's (Honnibal and Montani, 2017) dependency parser along with rule-based methods to identify each configuration.

We observe that the end-to-end baseline model generates much longer text (both number of words and sentences). Manual inspection shows that it introduces repetitions that are avoided in the planner-realizer approach. Another notable finding is that the frequencies of cohesive devices in texts generated by the planner-realizer model are much more similar to those in the reference texts than the end-to-end approach.

In this study, we illustrated a computational approach to justifying a modular approach for data to text generation. Starting with a very simple representation model of plans (as a sequence of groups of triplets), we specified a learnable text plan module and an evaluation metrics for the generated plans. We demonstrated empirically that a modular model separating planning and realization generates more cohesive text with less repetitions. We believe this empirical platform opens routes for fruitful exchange with more sophisticated text planning models and their interaction with realization.

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.

Claire Gardent, Anastasia Shimorina, Shashi Narayan, and Laura Perez-Beltrachini. 2017a. Creating training corpora for NLG micro-planners. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 179–188, Vancouver, Canada. Association for Computational Linguistics.

Claire Gardent, Anastasia Shimorina, Shashi Narayan, and Laura Perez-Beltrachini. 2017b. The WebNLG challenge: Generating text from RDF data. In *Proceedings of the 10th International Conference on Natural Language Generation*, pages 124–133, Santiago de Compostela, Spain. Association for Computational Linguistics.

Albert Gatt and E. Krahmer. 2018. Survey of the state of the art in natural language generation: Core tasks, applications and evaluation. *JAIR*, 61.

M. A. K. Halliday and R. Hasan. 1976. *Cohesion in English*. Longman.

Matthew Honnibal and Ines Montani. 2017. spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing. To appear.

Mihir Kale. 2020. Text-to-text pre-training for data-to-text tasks.

Jan Van Kuppevelt. 1996. Directionality in Discourse: Prominence Differences in Subordination Relations. *Journal of Semantics*, 13(4):363–395.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Amit Moryossef, Yoav Goldberg, and Ido Dagan. 2019. 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.

Joakim Nivre, Johan Hall, and Jens Nilsson. 2004. Memory-based dependency parsing. In *Proceedings of the Eighth Conference on Computational Natural Language Learning (CoNLL-2004) at HLT-NAACL 2004*, pages 49–56, Boston, Massachusetts, USA. Association for Computational Linguistics.

Craige Roberts. 2012. Information structure in discourse: Towards an integrated formal theory of pragmatics. *Semantics and Pragmatics*, 5(6):1–69.