Data-to-Text Generation with Iterative Text Editing

Zdeněk Kasner and **Ondřej Dušek**
Charles University, Faculty of Mathematics and Physics
Institute of Formal and Applied Linguistics
Prague, Czech Republic
{kasner,odusek}@ufal.mff.cuni.cz

Abstract

We present a novel approach to data-to-text generation based on iterative text editing. Our approach maximizes the completeness and semantic accuracy of the output text while leveraging the abilities of recent pre-trained models for text editing (LASERTAGGER) and language modeling (GPT-2) to improve the text fluency. To this end, we first transform data items to text using trivial templates, and then we iteratively improve the resulting text by a neural model trained for the *sentence fusion* task. The output of the model is filtered by a simple heuristic and reranked with an off-the-shelf pre-trained language model. We evaluate our approach on two major data-to-text datasets (WebNLG, Cleaned E2E) and analyze its caveats and benefits. Furthermore, we show that our formulation of data-to-text generation opens up the possibility for zero-shot domain adaptation using a general-domain dataset for sentence fusion. [1]

1 Introduction

Data-to-text (D2T) generation is the task of transforming structured data into a natural language text which represents it (Reiter and Dale, 2000; Gatt and Krahmer, 2018). The output text can be generated in several steps following a pipeline, or in an end-to-end (E2E) fashion. Neural-based E2E architectures recently gained attention due to their potential to reduce the human input needed for building D2T systems. A disadvantage of E2E architectures is the lack of intermediate steps, which makes it hard to control the semantic fidelity of the output (Moryossef et al., 2019b; Castro Ferreira et al., 2019).

We focus on a D2T setup where the input data is a set of RDF triples in the form of *(subject, predicate, object)* and the output text represents *all* and *only* facts in the data. This setup can be used by all D2T applications where the data describe relationships between entities (e.g. Gardent et al., 2017; Budzianowski et al., 2018).[2] In order to combine the benefits of pipeline and E2E architectures, we propose to use the neural models with a limited scope. We take advantage of three facts: (1) each triple can be lexicalized using a trivial template, (2) stacking the lexicalizations one after another tends to produce an unnatural sounding but semantically accurate output, and (3) the neural model can be used for combining the lexicalizations to improve the output fluency.

In traditional pipeline-based NLG systems (Reiter and Dale, 2000), combining the lexicalizations is a non-trivial multi-stage process. Text structuring and sentence aggregation are first used to determine the order of facts and their assignment to sentences, followed by referring expression generation and linguistic realization. We argue that with a neural model, combining the lexicalizations can be simplified as several iterations of *sentence fusion*—a task of combining sentences into a coherent text (Barzilay and McKeown, 2005).

Our contributions are the following:

1) We show how to reframe D2T generation as iterative text editing, which makes it independent of dataset-specific input data format and allows to control the output over a series of intermediate steps.
2) We perform initial experiments using our approach on two major D2T datasets (WebNLG and Cleaned E2E) and include a quantitative and qualitative analysis of the results.
3) We perform zero-shot domain adaptation experiments and show that our approach exhibits a domain-independent behavior.

---

[1] The code for the experiments is available at https://github.com/kasnerz/d2t_iterative_editing

[2] The setup can be preceded by the *content selection* for selecting the relevant subset of data (cf. Wiseman et al., 2017).
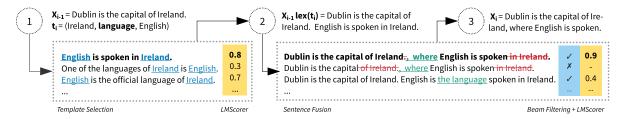
Figure 1: An example of a single iteration of our algorithm for D2T generation. In Step 1, the template for the triple is selected and filled. In Step 2, the sentence is fused with the template. In Step 3, the result for the next iteration is selected from the beam by filtering and language model scoring.

2 Background

Improving the accuracy of neural D2T approaches has attracted a lot of research interest lately. Similarly to us, other systems use a generate-then-rerank approach (Dušek and Jurčíček, 2016; Juraska et al., 2018) or a classifier to filter incorrect output (Harkous et al., 2020). Moryossef et al. (2019a,b) split the D2T process into a symbolic text-planning stage and a neural generation stage. Other works improve the robustness of the neural model (Tian et al., 2019; Kedzie and McKeown, 2019) or employ a natural language understanding model (Nie et al., 2019) to improve the faithfulness of the output. Recently, Chen et al. (2020) fine-tuned GPT-2 (Radford et al., 2019) for a few-shot domain adaptation.

Several models were recently applied to generic text editing tasks. LASERTAGGER (Malmi et al., 2019), which we use in our approach, is a sequence tagging model based on the Transformer (Vaswani et al., 2017) architecture with the BERT (Devlin et al., 2019) pre-trained language model as the encoder. Other recent text-editing models without a pre-trained backbone include EditNTS (Dong et al., 2019) and Levenshtein Transformer (Gu et al., 2019).

Concurrently with our work, Kale and Rastogi (2020) explored using templates for dialogue response generation. They use the sequence-to-sequence T5 model (Raffel et al., 2019) to generate the output text from scratch instead of iteratively editing the intermediate outputs, which leaves less control over the model.

3 Our Approach

We start from single-triple templates and iteratively fuse them into the resulting text while filtering and reranking the results. We first detail the main components of our system and then give an overall description of the decoding algorithm.

3.1 Template Extraction

We collect a set of templates for each predicate. The templates can be either handcrafted, or automatically extracted from the lexicalizations of the single-triple examples in the training data. For unseen predicates, we add a single fallback template: *The <predicate> of <subject> is <object>*.

3.2 Sentence Fusion

We train an in-domain sentence fusion model. We select pairs (X, X') of examples from the training data consisting of $(n, n + 1)$ triples and having n triples in common. This leaves us with an extra triple t present only in X'. To construct the training data, we use the concatenated sequence $X \operatorname{lex}(t)$ as a source and the sequence X' as a target, where $\operatorname{lex}(t)$ denotes lexicalizing the triple t using an appropriate template. As a result, the model learns to integrate X and t into a single coherent expression.

We base our sentence fusion model on LASERTAGGER (Malmi et al., 2019). LASERTAGGER is a sequence generation model which generates outputs by tagging inputs with edit operations: KEEP a token, DELETE a token, and ADD a phrase before the token. In tasks where the output highly overlaps with the input, such as sentence fusion, LASERTAGGER is able to achieve performance comparable to state-of-the-art models with faster inference times and less training data.

An important feature of LASERTAGGER is the limited size of its vocabulary, which consists of l most frequent (possibly multi-token) phrases used to transform inputs to outputs in the training data. After the vocabulary is precomputed, all infeasible examples in the training data are filtered out. At the cost of limiting the number of training examples, this filtering makes the training data cleaner by removing outliers. The limited vocabulary also

	WebNLG
foundedBy	<obj> was the founder of <subj>. <subj> was founded by <obj>.
	E2E (extracted)
area+food	<subj> offers <obj2> cuisine in the <obj1>. <subj> in <obj1> serves <obj2> food.
	E2E (custom)
near	<subj> is located near <obj>. <obj> is close to <subj>.

Table 1: Examples of templates we used in our experiments. The templates for the single predicates in the WebNLG dataset and the pairs of predicates in the E2E dataset are extracted automatically from the training data; the templates for the single predicates in E2E are created manually.

makes the model less prone to common neural model errors such as hallucination, which allows us to control the semantic accuracy to a great extent using only simple heuristics and language model rescoring.

3.3 LM Scoring

We use an additional component for calculating an indirect measure of the text fluency. We refer to the component as the LMSCORER. In our case, LM-SCORER is a pre-trained GPT-2 language model (Radford et al., 2019) from the Transformers repository[3] (Wolf et al., 2019) wrapped in the *lm-scorer*[4] package. We use LMSCORER to compute the score of the input text X composed of tokens $x_1 \ldots x_n$ as a geometric mean of the token conditional probability:

$$\text{score}(X) = \left(\prod_{i=1}^{n} P(x_i | x_1 \ldots x_{i-1}) \right)^{\frac{1}{n}}.$$

3.4 Decoding Algorithm

The input of the algorithm (Figure 1) is a set of n ordered triples. First, we lexicalize the triple t_0 to get the base text X_0. We choose the lexicalization for the triple as the filled template with the best score from LMSCORER. This promotes templates which sound more natural for particular values. In the following steps $i = 1 \ldots n$, we lexicalize the triple t_i and append it after X_{i-1}. We feed the joined text into the sentence fusion model and produce a beam with fusion hypotheses. We use a simple

heuristic (string matching) to filter out hypotheses in the beam missing any entity from the input data. Finally, we rescore the remaining hypotheses in the beam with LMSCORER and let the hypothesis with the best score be the base text X_i. In case there are no sentences left in the beam after the filtering step, we let X_i be the text in which the lexicalized t_i is appended after X_{i-1} without fusion (preferring accuracy to fluency). The output of the algorithm is the base text X_n from the final step.

4 Experiments

4.1 Datasets

The WebNLG dataset (Gardent et al., 2017) consists of sets of DBPedia RDF triples and their lexicalizations. Following previous work, we use version 1.4 from Castro Ferreira et al. (2018). The E2E dataset (Novikova et al., 2017) contains restaurant descriptions based on sets of attributes (slots). In this work, we refer to the cleaned version of the E2E dataset (Dušek et al., 2019). For the domain adaptation experiments, we use DISCOFUSE (Geva et al., 2019), which is a large-scale dataset for sentence fusion.

4.2 Data Preprocessing

For WebNLG, we extract the initial templates from the training data from examples containing only a *single* triple. In the E2E dataset, there are no such examples; therefore our solution is twofold: we extract the templates for *pairs* of predicates, using them as a starting point for the algorithm in order to leverage the lexical variability in the data (manually filtering out the templates with semantic noise), and we also create a small set of templates for each *single* predicate manually, using them in the subsequent steps of the algorithm (this is possible due to the low variability of the predicates in the dataset).[5] See Table 1 for examples of templates we used in our experiments.

4.3 Setup

As a *baseline*, we generate the best templates according to LMSCORER without applying the sentence fusion (i.e. always using the fallback).

For the *sentence fusion* experiments, we use LASERTAGGER with the autoregressive decoder

---

[3]https://github.com/huggingface/transformers

[4]https://github.com/simonepri/lm-scorer

---

[5]In the E2E dataset, the data is in the form of key-value slots. We transform the data to RDF triples by using the name of the restaurant as a *subject* and the rest of the slots as *predicate* and *object*. This creates *n-1* triples for *n* slots.

	WebNLG					Cleaned E2E				
	BLEU	*NIST*	*METEOR*	*ROUGE$_L$*	*CIDEr*	*BLEU*	*NIST*	*METEOR*	*ROUGE$_L$*	*CIDEr*
baseline	0.277	6.328	0.379	0.524	1.614	0.207	3.679	0.334	0.401	0.365
zero-shot	0.288	6.677	0.385	0.530	1.751	0.220	3.941	0.340	0.408	0.473
w/fusion	0.353	7.923	0.386	0.555	2.515	0.252	4.460	0.338	0.436	0.944
SFC	0.524	-	0.424	0.660	3.700	0.436	-	0.390	0.575	2.000
T5	0.571	-	0.440	-	-	-	-	-	-	-

Table 2: Results of automatic metrics on the WebNLG and Cleaned E2E test sets. The comparison is made with the results from the papers on the Semantic Fidelity Classifier (SFC; Harkous et al., 2020) and the finetuned T5 model (T5; Kale, 2020).

with a beam of size 10. We use all reference lexicalizations and the vocabulary size $V = 100$, following our preliminary experiments, which showed that filtering the references only by limiting the vocabulary size brings the best results (see Supplementary for details). We finetune the model for 10,000 updates with batch size 32 and learning rate 2×10^{-5}. For the beam filtering heuristic, we check for the presence of entities by simple string matching in WebNLG; for the E2E dataset, we use a set of regular expressions from TGen[6] (Dušek et al., 2019). We do not use any pre-ordering steps for the triples and process them in the default order.

Additionally, we conduct a *zero-shot domain adaptation* experiment. We train the sentence fusion model with the same setup, but instead of the in-domain datasets, we use a subset of the balanced-Wikipedia portion of the DISCOFUSE dataset. In particular, we use the discourse types which frequently occur in our datasets, filtering the discourse types which are not relevant for our use-case. See Supplementary for the full listing of the selected types.

5 Analysis of Results

We compute the metrics used in the evaluation of the E2E Challenge (Dušek et al., 2020): BLEU (Papineni et al., 2002), NIST (Doddington, 2002), METEOR (Banerjee and Lavie, 2005), ROUGE$_L$ (Lin, 2004) and CIDEr (Vedantam et al., 2015). The results are shown in Table 2. The scores from the automatic metrics lag behind the state-of-the-art, although both the fusion and the zero-shot approaches show improvements over the baseline. We examine the details in the following paragraphs, discussing the behavior of our approach, and we outline plans for improving the results in Section 6.

Accuracy vs. Variability Our approach ensures zero entity errors, since the entities are filled ver-

batim into the templates and in case an entity is missing in the whole beam, a fallback is used instead. Semantic inconsistencies still occur, e.g. if a verb or function words are missing.

The fused sentences in the E2E dataset, where all the objects are related to a single subject, often lean towards compact forms, e.g.: *Aromi is a family friendly chinese coffee shop with a low customer rating in riverside*. On the contrary, the sentence structure in WebNLG mostly follows the structure from the templates and the model performs minimal changes to fuse the sentences together. See Table 3 and Supplementary for examples of the system outputs. Out of all steps, 28% are fallbacks (no fusion is performed) in WebNLG and 54% in the E2E dataset. The higher number of fallbacks in the E2E dataset can be explained by a higher lexical variability of the references, together with a higher number of data items per example in the E2E dataset, making it harder for the model to maintain the text coherency over multiple steps.

Templates On average, there are 12.4 templates per predicate in WebNLG and 8.3 in the E2E dataset. In cases where the set of templates is more diverse, e.g. if the template for the predicate *country* has to be selected from {*<subject> is situated within <object>, <subject> is a dish found in <object>*}, LMSCORER helps to select the semantically accurate template for the specific entities. The literal copying of entities can be too rigid in some cases, e.g. *Atatürk Monument (İzmir) is made of "Bronze"*, but these disfluencies can be improved in the fusion step.

Reordering LASERTAGGER does not allow arbitrary reordering of words in the sentence, which can limit the expressiveness of the sentence fusion model. Consider the example in Figure 1: in order to create a sentence *English is spoken in Dublin, the capital of Ireland*, the model has to delete and re-insert at least one of the entities, e.g. *English*,

---
[6] https://github.com/UFAL-DSG/tgen

Triples	*(Albert Jennings Fountain, deathPlace, New Mexico Territory); (Albert Jennings Fountain, birthPlace, New York City); (Albert Jennings Fountain, birthPlace, Staten Island)*
Step #0	Albert Jennings Fountain died in New Mexico Territory.
Step #1	Albert Jennings Fountain, who died in New Mexico Territory, was born in New York City.
Step #2	Albert Jennings Fountain, who died in New Mexico Territory, was born in New York City, Staten Island.
Reference	Albert Jennings Fountain was born in Staten Island, New York City and died in the New Mexico Territory.

Table 3: An example of correct behavior of the algorithm on the WebNLG dataset. Newly added entities are underlined, the output from Step #2 is the output text.

which has to be present in the vocabulary.

Domain Independence The zero-shot model trained on DISCOFUSE is able to correctly pronominalize or delete repeated entities and join the sentences with conjunctives, e.g. *William Anders was born in British Hong Kong, and was a member of the crew of Apollo 8.* While the model makes only a limited use of sentence fusion, it makes the output more fluent while keeping strong guarantees of the output accuracy.

6 Future Work

Although the current version of our approach is not yet able to consistently produce sentences with a high degree of fluency, we believe that the approach provides a valuable starting point for controllable and domain-independent D2T generation. In this section, we outline possible directions for tackling the main drawbacks and improving the results of the model with further research.

Building a high-quality sentence fusion model, which lies at the core of our approach, remains a challenge (Lebanoff et al., 2020). Our simple extractive approach relying on existing D2T datasets may not produce sufficient amount of clean data. On the other hand, the phenomena covered in the DISCOFUSE dataset are too narrow for the fully general sentence fusion. We believe that training the sentence fusion model on a larger and more diverse sentence fusion dataset, built e.g. in an unsupervised fashion (Lebanoff et al., 2019), is a way to improve the robustness of our approach.

Fluency of the output sentences may be also improved by allowing more flexibility for the order of entities, either by including an ordering step in the pipeline (Moryossef et al., 2019b), or by using a text-editing model that is capable of explicit reordering of words in the sentence (Mallinson et al., 2020). Splitting the data into smaller batches (i.e. setting an upper bound for the number of sentences fused together) could also help to improve the consistency of results with a higher number of data items.

Our string matching heuristic is quite crude and may lead to a high number of fallbacks. Introducing a more precise heuristic, such as a semantic fidelity classifier (Harkous et al., 2020), or a model trained for natural language inference (Dušek and Kasner, 2020) could help to promote lexical variability of the text.

Finally, we note that the text-editing paradigm allows to visualize the changes made by the model, introducing the option to accept or reject the changes at each step, and even build a set of custom rules on top of the individual edit operations based on the affected tokens. This flexibility could be useful for tweaking the model manually for a production system.

7 Conclusions

We proposed a simple and intuitive approach for D2T generation, splitting the process into two steps: lexicalization of data and improving the text fluency. A trivial lexicalization helps to promote fidelity and domain independence while delegating the subtle work with language to neural models allows to benefit from the power of general-domain pre-training. While a straightforward application of this approach on the WebNLG and E2E datasets does not produce state-of-the-art results in terms of automatic metrics, the results still show considerable improvements above the baseline. We provided insights into the behavior of the model, highlighted its potential benefits, and proposed the directions for further improvements.

Acknowledgements

References

Satanjeev Banerjee and Alon Lavie. 2005. 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*, pages 65–72, Ann Arbor, Michigan. Association for Computational Linguistics.

Regina Barzilay and Kathleen R. McKeown. 2005. Sentence fusion for multidocument news summarization. *Computational Linguistics*, 31(3):297–328.

Paweł Budzianowski, Tsung-Hsien Wen, Bo-Hsiang Tseng, Iñigo Casanueva, Stefan Ultes, Osman Ramadan, and Milica Gašić. 2018. MultiWOZ - a large-scale multi-domain Wizard-of-Oz dataset for task-oriented dialogue modelling. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 5016–5026, Brussels, Belgium. Association for Computational Linguistics.

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. Association for Computational Linguistics.

Thiago Castro Ferreira, Diego Moussallem, Emiel Krahmer, and Sander Wubben. 2018. Enriching the WebNLG corpus. In *Proceedings of the 11th International Conference on Natural Language Generation*, pages 171–176. Association for Computational Linguistics.

Zhiyu Chen, Harini Eavani, Wenhu Chen, Yinyin Liu, and William Yang Wang. 2020. Few-shot NLG with pre-trained language model. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 183–190, Online. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. 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 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

George Doddington. 2002. Automatic evaluation of machine translation quality using n-gram co-occurrence statistics. In *Proceedings of the Second International Conference on Human Language Technology Research*, pages 138–145.

Yue Dong, Zichao Li, Mehdi Rezagholizadeh, and Jackie Chi Kit Cheung. 2019. EditNTS: An neural programmer-interpreter model for sentence simplification through explicit editing. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3393–3402, Florence, Italy. Association for Computational Linguistics.

Ondřej Dušek, Jekaterina Novikova, and Verena Rieser. 2020. Evaluating the state-of-the-art of end-to-end natural language generation: The E2E NLG Challenge. *Computer Speech & Language*, 59:123–156.

Ondřej Dušek, David M. Howcroft, and Verena Rieser. 2019. Semantic noise matters for neural natural language generation. In *Proceedings of the 12th International Conference on Natural Language Generation*, pages 421–426, Tokyo, Japan. Association for Computational Linguistics.

Ondřej Dušek and Filip Jurčíček. 2016. Sequence-to-sequence generation for spoken dialogue via deep syntax trees and strings. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 45–51, Berlin. Association for Computational Linguistics. ArXiv:1606.05491.

Ondřej Dušek and Zdeněk Kasner. 2020. Evaluating semantic accuracy of data-to-text generation with natural language inference. In *Proceedings of the 13th International Conference on Natural Language Generation*. Association for Computational Linguistics.

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

Albert Gatt and Emiel Krahmer. 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.

Mor Geva, Eric Malmi, Idan Szpektor, and Jonathan Berant. 2019. DiscoFuse: A large-scale dataset for discourse-based sentence fusion. 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 3443–3455, Minneapolis, Minnesota. Association for Computational Linguistics.

Jiatao Gu, Changhan Wang, and Junbo Zhao. 2019. Levenshtein transformer. In *Advances in Neural Information Processing Systems*, pages 11181–11191.

Hamza Harkous, Isabel Groves, and Amir Saffari. 2020. Have your text and use it too! End-to-end neural data-to-text generation with semantic fidelity. *arXiv preprint arXiv:2004.06577*.

Juraj Juraska, Panagiotis Karagiannis, Kevin Bowden, and Marilyn Walker. 2018. A deep ensemble model with slot alignment for sequence-to-sequence natural language generation. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 152–162, New Orleans, LA, USA. Association for Computational Linguistics.

Mihir Kale. 2020. Text-to-text pre-training for data-to-text tasks. *arXiv preprint arXiv:2005.10433*.

Mihir Kale and Abhinav Rastogi. 2020. Few-shot natural language generation by rewriting templates. *arXiv preprint arXiv:2004.15006*.

Chris Kedzie and Kathleen McKeown. 2019. A good sample is hard to find: Noise injection sampling and self-training for neural language generation models. In *Proceedings of the 12th International Conference on Natural Language Generation*, pages 584–593, Tokyo, Japan. Association for Computational Linguistics.

Logan Lebanoff, Franck Dernoncourt, Doo Soon Kim, Lidan Wang, Walter Chang, and Fei Liu. 2020. Learning to fuse sentences with transformers for summarization. *arXiv preprint arXiv:2010.03726*.

Logan Lebanoff, Kaiqiang Song, Franck Dernoncourt, Doo Soon Kim, Seokhwan Kim, Walter Chang, and Fei Liu. 2019. Scoring sentence singletons and pairs for abstractive summarization. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2175–2189, Florence, Italy. Association for Computational Linguistics.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Jonathan Mallinson, Aliaksei Severyn, Eric Malmi, and Guillermo Garrido. 2020. Felix: Flexible text editing through tagging and insertion. *arXiv preprint arXiv:2003.10687*.

Eric Malmi, Sebastian Krause, Sascha Rothe, Daniil Mirylenka, and Aliaksei Severyn. 2019. Encode, tag, realize: High-precision text editing. 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 5054–5065. Association for Computational Linguistics.

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. 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. Association for Computational Linguistics.

Feng Nie, Jin-Ge Yao, Jinpeng Wang, Rong Pan, and Chin-Yew Lin. 2019. A simple recipe towards reducing hallucination in neural surface realisation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2673–2679, Florence, Italy. Association for Computational Linguistics.

Jekaterina Novikova, Ondřej Dušek, and Verena Rieser. 2017. The E2E dataset: New challenges for end-to-end generation. In *Proceedings of the 18th Annual SIGdial Meeting on Discourse and Dialogue*, pages 201–206, Saarbrücken, Germany. Association for Computational Linguistics.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 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, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language Models are Unsupervised Multitask Learners. Technical report, OpenAI.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv preprint arXiv:1910.10683*.

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

Ran Tian, Shashi Narayan, Thibault Sellam, and Ankur P Parikh. 2019. Sticking to the facts: Confident decoding for faithful data-to-text generation. *arXiv preprint arXiv:1910.08684*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Ramakrishna Vedantam, C Lawrence Zitnick, and Devi Parikh. 2015. CIDEr: Consensus-based image description evaluation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4566–4575.

Sam Wiseman, Stuart Shieber, and Alexander Rush. 2017. Challenges in data-to-document generation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2253–2263, Copenhagen, Denmark. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, R'emi Louf, Morgan Funtowicz, and Jamie Brew. 2019. Huggingface's transformers: State-of-the-art natural language processing. *ArXiv*, abs/1910.03771.

Data-to-Text Generation with Iterative Text Editing: Supplementary Material

A Hyperparameter Setup

Examples in the original datasets can have multiple reference lexicalizations. We introduce three strategies for dealing with this fact during the construction of the training dataset for the sentence fusion model:

- *"best"*: select the best lexicalizations for both the source and the target using LMSCORER
- *"best_tgt"*: select the best lexicalization for the target using LMSCORER and use all lexicalizations for the source
- *"all"*: use all lexicalizations for both the source and the target

Note that the training dataset is further filtered by the limited vocabulary of LASERTAGGER, which helps to filter out the outliers. We experiment with vocabulary sizes $V \in \{100, 500, 1000, 5000\}$. Table 4 shows the results on the development sets of both datasets. Based on these results, we select $V = 100$ and the strategy *all* for our final experiments.

		WebNLG										
	best				*best_tgt*				*all*			
vocab. size	100	500	1000	5000	100	500	1000	5000	100	500	1000	5000
BLEU	0.373	0.370	0.370	0.335	0.382	0.382	0.375	0.342	**0.397**	0.389	0.391	0.370
NIST	7.610	7.478	7.411	6.713	7.673	7.596	7.470	6.831	**7.912**	7.676	7.679	7.307
METEOR	0.398	0.399	0.397	0.396	**0.401**	0.399	0.396	0.393	0.400	**0.401**	0.400	0.399
ROUGE$_L$	0.566	0.569	0.568	0.553	0.569	0.570	0.569	0.556	0.574	**0.577**	0.576	0.568
CIDER	2.586	2.573	2.466	2.023	2.594	2.525	2.466	2.133	**2.639**	2.570	2.557	2.385

		E2E										
	best				*best_tgt*				*all*			
vocab. size	100	500	1000	5000	100	500	1000	5000	100	500	1000	5000
BLEU	0.252	0.254	0.249	0.255	0.269	0.258	0.260	0.256	**0.293**	0.277	0.273	0.268
NIST	4.168	4.180	4.049	4.077	4.435	4.167	4.154	4.097	**4.762**	4.461	4.357	4.238
METEOR	0.345	0.346	0.348	0.351	0.351	0.352	0.351	0.350	0.353	0.350	0.352	**0.355**
ROUGE$_L$	0.426	0.435	0.429	0.429	0.441	0.434	0.435	0.430	**0.460**	0.448	0.447	0.441
CIDEr	0.739	0.759	0.647	0.634	0.929	0.728	0.693	0.678	**1.128**	0.967	0.881	0.799

Table 4: Results of automatic metrics on the WebNLG and E2E development sets with different reference strategies and vocabulary sizes.

B Discourse Types

The list of different discourse types available in the DISCOFUSE dataset, with an indication whether they were selected for our zero-shot training, is shown in Table 5.

type	selected	type	selected
PAIR_ANAPHORA	yes	SINGLE_CONN_INNER_ANAPHORA	no
PAIR_CONN	no	SINGLE_CONN_START	no
PAIR_CONN_ANAPHORA	no	SINGLE_RELATIVE	yes
PAIR_NONE	yes	SINGLE_S_COORD	yes*
SINGLE_APPOSITION	yes	SINGLE_S_COORD_ANAPHORA	yes*
SINGLE_CATAPHORA	no	SINGLE_VP_COORD	yes*
SINGLE_CONN_INNER	no		

Table 5: A list of available discourse types in the DISCOFUSE dataset. For our zero-shot experiments, we select a subset of DISCOFUSE, omitting the phenomena which mostly do not occur in our datasets. The asterisk (*) symbolizes that only the examples with the connectives *"and"* or *", and"* were selected.

C Output Examples

Tables 7–8 show examples of outputs of our iterative sentence fusion method (with in-domain training) on both the E2E and WebNLG datasets. We show both instances that produce flawless output (Tables 6 and 7) and instances where our approach makes an error (Table 8 and 9). Table 10 then illustrates the behavior of the zero-shot approach (without in-domain training data).

Triples	(A Loyal Character Dancer, publisher, Soho Press); (Soho Press, country, United States); (United States, leaderName, Barack Obama)
Step #0	Soho Press is the publisher of A Loyal Character Dancer.
Step #1	Soho Press is the publisher of A Loyal Character Dancer which can be found in the United States.
Step #2	Soho Press is the publisher of A Loyal Character Dancer which can be found in the United States where Barack Obama is president.
Reference	A Loyal Character Dancer is published by Soho Press in the United States where Barack Obama is the president.

Table 6: An example of correct behavior of the algorithm on the WebNLG dataset (newly added entities are underlined).

Triples	(Giraffe, area, riverside); (Giraffe, eatType, pub); (Giraffe, familyFriendly, no); (Giraffe, food, Chinese); (Giraffe, near, Raja Indian Cuisine)
Step #0	Giraffe serves French food and is not family-friendly. ↳ *A template for the pair of predicates "eatType" and "familyFriendly" is selected.*
Step #1	Giraffe serves French food in the riverside area and is not family-friendly.
Step #2	Giraffe is a French pub in the riverside area that is not family-friendly.
Step #3	Giraffe is a French pub in riverside that is not family-friendly. It is located near Raja Indian Cuisine .
Reference	Giraffe is a not family-friendly French pub near Raja Indian Cuisine near the riverside.

Table 7: An example of correct behavior of the algorithm on the E2E dataset (newly added entities are underlined).

Triples	(Poland, language, Polish language); (Adam Koc, nationality, Poland); (Poland, ethnicGroup, Kashubians)
Step #0	Polish language is one of the languages that is spoken in Poland .
Step #1	Polish language is spoken in Poland, where Adam Koc is spoken. ↳ *An incorrect expression is inserted.*
Step #2	Polish language is spoken in Poland, where Adam Koc is spoken and Kashubians are an ethnic group.
Reference	The Polish language is used in Poland, where Adam koc was from. Poland has an ethnic group called Kashubians.

Table 8: An example of incorrect behavior of the algorithm on the WebNLG dataset (with the error underlined).

Triples	(The Phoenix, area, riverside); (The Phoenix, eatType, restaurant); (The Phoenix, familyFriendly, yes); (The Phoenix, near, Raja Indian Cuisine); (The Phoenix, priceRange, cheap)
Step #0	The Phoenix is a cheap place to eat. Yes it is family friendly.
	↳ *A template for the pair of predicates "price" and "familyFriendly" is selected.*
Step #1	The Phoenix is a cheap family friendly on the riverside.
	↳ *A grammatical error is made.*
Step #2	The Phoenix is a cheap family friendly offering restaurant in the riverside area.
	↳ *The grammar of the sentence is still not correct.*
Step #3	The Phoenix is a cheap, family friendly restaurant in the riverside area, located near Raja Indian Cuisine.
	↳ *Grammatical errors are fixed in the last step of sentence fusion.*
Reference	Cheap food and a family friendly atmosphere at The Phoenix restaurant. Situated riverside near the Raja Indian Cuisine.

Table 9: An example of behavior of the algorithm on the E2E dataset with several intermediate mistakes (underlined) and fixed output.

Triples	(Arrabbiata sauce, region, Rome); (Arrabbiata sauce, country, Italy); (Arrabbiata sauce, ingredient, olive oil)
Step #0	Arrabbiata sauce is a dish that comes from the Rome region.
	↳ *A template for the predicate "region" (suitable for food) is selected.*
Step #1	Arrabbiata sauce is a dish that comes from the Rome region, and it is a dish that is popular in Italy.
	↳ *The sentences are correctly joined together.*
Step #2	Arrabbiata sauce is a dish that comes from the Rome region, and it is a dish that is popular in Italy. Olive oil is one of the ingredients used to make Arrabbiata sauce.
	↳ *The text is left intact.*
Reference	Arrabbiata sauce is a traditional dish from Rome, Italy. Olive oil is one of the ingredients in the sauce.

Table 10: An example of behavior of the zero-shot algorithm on the WebNLG dataset (with a single change made by the editing step underlined).