Text-in-Context: Token-Level Error Detection for Table-to-Text Generation

Zdeněk Kasner,[1] **Simon Mille**[2] and **Ondřej Dušek**[1]
[1]Charles University, Faculty of Mathematics and Physics, Prague, Czech Republic
[2]Pompeu Fabra University, Barcelona, Spain
kasner@ufal.mff.cuni.cz, simon.mille@upf.edu, odusek@ufal.mff.cuni.cz

Abstract

We present our Charles-UPF submission for the Shared Task on Evaluating Accuracy in Generated Texts at INLG 2021. Our system can detect the errors automatically using a combination of a rule-based natural language generation (NLG) system and pretrained language models (LMs). We first utilize a rule-based NLG system to generate sentences with facts that can be derived from the input. For each sentence we evaluate, we select a subset of facts which are relevant by measuring semantic similarity to the sentence in question. Finally, we finetune a pretrained language model on annotated data along with the relevant facts for fine-grained error detection. On the test set, we achieve 69% recall and 75% precision with a model trained on a mixture of human-annotated and synthetic data.

1 Introduction

Recent neural NLG systems can easily generate fluent texts from linearized structured data (Zhao et al., 2020; Kale and Rastogi, 2020; Castro Ferreira et al., 2020). However, the systems cannot guarantee that the output is properly grounded in the input – hallucination (outputs not supported by input data) is a notorious problem in neural NLG (Tian et al., 2019; Harkous et al., 2020; Filippova, 2020; Rebuffel et al., 2021). Neural systems are particularly unreliable on complex datasets such as Rotowire (Wiseman et al., 2017), where the task is to generate basketball match summaries from tabular data. Rotowire poses multiple challenges for neural systems: it requires content selection and production of longer texts, and its human-written training texts are themselves not always grounded in data, which makes neural models more susceptible to hallucination.

On the other hand, rule-based systems used in recent data-to-text tasks (Lapalme, 2020; Tran and Nguyen, 2020; Mille et al., 2019) all achieve high scores in terms of accuracy of the generated contents with respect to the input structures (Dušek et al., 2020; Castro Ferreira et al., 2020). This, however, comes with the cost of lower fluency.

Detecting NLG errors automatically is a hard problem. For word-overlap-based metrics, such as BLEU (Papineni et al., 2002) or METEOR (Lavie and Agarwal, 2007), reliability on content checking is known to be poor (Novikova et al., 2017; Dhingra et al., 2019). Most neural metrics (Zhang et al., 2020; Sellam et al., 2020) have not been evaluated for content preservation. Dušek and Kasner (2020)'s metric based on natural language inference (NLI) specifically targets content preservation, but, same as all previously mentioned ones, is not able to provide fine-grained error tagging beyond sentence level. Specific content-checking metrics mostly remain a domain of handcrafted pattern matching (Wen et al., 2015; Dušek et al., 2019), which does not scale well to new domains. While human evaluation provides a more reliable alternative, it is costly and difficult to set up (van der Lee et al., 2019; Santhanam and Shaikh, 2019; Belz et al., 2020; Thomson and Reiter, 2020a).

The INLG 2021 accuracy evaluation shared task (Reiter and Thomson, 2020; Thomson and Reiter, 2021) aims to improve this situation. Reiter and Thomson (2020) carefully annotated 60 outputs of various neural systems trained on Rotowire with 6 error types (see Table 1) defined in Thomson and Reiter (2020b). The objective of the shared task is then to either implement an automatic metric for creating the same type of annotations automatically, or to develop a human evaluation scenario capable of producing the same annotations while requiring less resources.

Our submission for the shared task falls into the first category: we developed an automatic metric for token-level error annotation which combines a

Proceedings of the 14th International Conference on Natural Language Generation (INLG), pages 259–265,
Aberdeen, Scotland, UK, 20-24 September 2021.

NUMBER	Incorrect number.
NAME	Incorrect named entity.
WORD	Any other incorrect word.
CONTEXT	A phrase inappropriate for the context.
NOT_CHECKABLE	A statement which cannot be checked.
OTHER	Any other type of mistake.

Table 1: Error categories for the Rotowire dataset.

rule-based generation system with a neural retrieval model and a pretrained neural LM used for error tagging. We evaluated our approach in a cross-validation scenario to select the best configuration for the shared task. Overall, our system is able to reach 65% error detection F1 score and ranked first out of four automatic submissions in the shared task. The code for our experiments is freely available on Github.[1]

2 Our System

Our system is composed of 3 steps: A rule-based generator for fact descriptions (see Figure 1 and Section 2.1), a retrieval system for selecting facts relevant for a given sentence (Section 2.2), and a token-level error tagger based on the RoBERTa pretrained LM (Section 2.3). The latter two steps are summarized in Figure 2. The LM tagger is trained on examples provided by shared task organizers, as well as on synthetic data based on the Rotowire training set (Section 2.4).

2.1 Rule-based Fact Descriptions

We use rule-based systems to generate natural language descriptions of facts from the input tables, relating to all players and both teams. The facts are later supplied to the error-checking model for grounding the evaluated sentence (see Section 2.3). We experiment with both *simple* descriptions created by filling in sentence templates, and *compact* descriptions generated using a grammar-based system. The simple system produces about 569 facts/sentences for each game. The compact system generates about 112 sentences per game, i.e., 5 times less; the game descriptions contain the same amount of information but the individual sentences are more syntactically complex.

Facts generated For each game, we first generate every fact in the input table, i.e., 44 facts about the game (hosting team, visiting team, date converted to weekday) and so-called line-score objects (team name and statistics) and box-score objects (player name, player team, player starting position and their personal statistics).

Subsequently, we generate 85 further facts that can be inferred from the input table. These are based on reading the first 20 human-written summaries in the training data and finding frequently mentioned facts that can easily be derived from input, such as which team won and by how much, comparisons between the team and player raw data (e.g., *Team A dominated the defensive rebounding*, *Team A and Team B committed the same number of fouls*; *Player X was the (second) best scorer of the game/his team*), complex statistics (e.g., *Team A totaled X steals*; *Player X (almost) recorded a double-double*), or an interpretation of some numbers (e.g., *Team A came back in the 4th quarter; Team A was efficient/bad at shooting*).[2]

Simple descriptions are produced by a template-based system, with one template per fact. We handcrafted 129 sentence templates to cover all the facts described above. A sentence template looks like the following: "*[PLAYER_NAME] scored [PTS] points.*", where square brackets indicate variables that are instantiated with the corresponding input values (see Figure 1 for sample sentences).

Compact descriptions are produced by the FORGe system (Mille et al., 2019), which allows for the generation of more compact sentences by instantiating abstract (predicate-argument) templates instead of full sentences for each fact. For instance, the template for the points scored would be: *[PLAYER_NAME] ←A1 provide A2→ point NonCore→[PTS]*, where A1 and A2 denote the first and second arguments respectively, and NonCore a non-argumental relation. FORGe receives a series of instantiated templates and performs surface realization in several steps, by first aggregating the templates based on predicate and/or argument identity, and then structuring, linearizing and inflecting components of the sentences. The FORGe grammars were used off-the-shelf,[3] with cross-sentence referring expression generation deactivated so that each generated sentence can be used on its own. We manually crafted 98 abstract templates and added a description of the included

[1] https://github.com/kasnerz/accuracySharedTask_CUNI-UPF

[2] A number of mentioned facts could not be obtained from the Rotowire data, as for instance the player stats per quarter, a career high points total, whether a player is an all-star or not, or if a player scored the winning shot.

[3] Minor debugging was needed to cover some new contexts.

Figure 1: Rule-based NLG which we use to generate facts from the input data. The facts are used as an input to the error checking model (see Figure 2). We experiment with (a) simple hand-crafted templates and (b) compact sentences generated by the FORGe system.

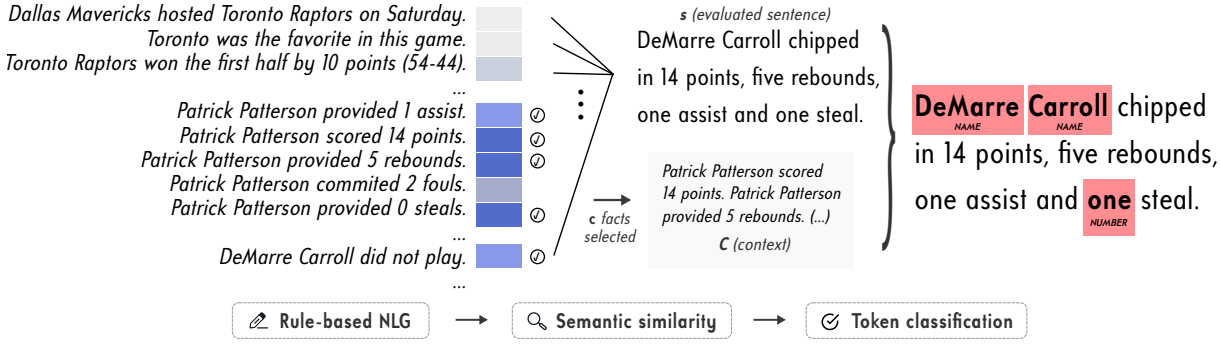

Figure 2: An overview of our system. First, we generate the facts from the input table with a rule-based NLG system (see Figure 1). For each evaluated sentence s, we select c facts with the highest semantic similarity, getting a context C. The pair (C, s) is given as an input to a pretrained LM for token-level error classification.

lexical units into the FORGe lexicon. For instance, the five simple sentences shown at the bottom of the yellow column in Figure 1 are covered by a single compact sentence shown at the bottom of the orange column.

2.2 Context Retrieval

Since the maximum length of the input sequence for our error-checking model (see Section 2.3) is 512 tokens (about 10% of the total length of the generated sentences G), we need to select only a subset of G, which we refer to as **context** C. We want to put the relevant sentences into C and filter out the rest to make the error tagging easier. This problem is hard in general, as any string matching (e.g. using numbers or names mentioned in the sentence) will fail on lexical variations.

Our solution is based on selecting sentences with the highest **semantic similarity**. For each generated sentence $g_i \in G$, we measure semantic similarity between g_i and the evaluated sentence s using Sentence Transformers (Reimers and Gurevych, 2019).[4] In particular, we embed the sentence tokens by applying mean pooling on the output of `paraphrase-distilroberta-base-v2`, getting the embedding vectors e_s and e_{g_i}. Then we compute the cosine similarity between the embeddings. For the context C, we select the top c sentences from G that have the highest cosine similarity to s.

2.3 LM-based Error Tagger

We use a RoBERTa LM (Liu et al., 2019) with a token-level classification head as our error-checking model. Unlike unsupervised approaches based on examining attention values (Thorne et al., 2019; Li et al., 2020) or input perturbations (Kim et al., 2020), we train the model directly to predict error categories using annotated data, similarly to Yoosuf and Yang (2019).

The model receives an input $X = (C, s)$, composed of the context C, i.e., relevant background facts selected by context retrieval in Section 2.2, and the generated sentence s to be tagged. The inputs are separated by the delimiter `</s>`. The model is trained to annotate each token in s either with an *OK* label, or with a label corresponding to one of the error categories.

We experiment with two data sources for training the model: (1) **gold-standard annotated data**

[4] https://www.sbert.net/

Generator	Data	c	EMR = 0.25			EMR = 0.5			EMR = 0.75		
			R	P	F1	R	P	F1	R	P	F1
Simple	synth	5	0.123	0.723	0.210	0.165	0.512	0.250	0.310	0.323	0.316
		10	0.138	0.737	0.232	0.181	0.549	0.272	0.328	0.400	0.360
		20	0.137	**0.741**	0.231	0.179	0.559	0.271	0.327	0.433	0.373
		40	0.165	0.712	0.268	0.199	0.560	0.294	0.296	0.436	0.353
	synth + human	5	0.422	0.617	0.501	0.414	0.594	0.488	0.401	0.583	0.475
		10	0.467	0.551	0.506	0.438	0.638	0.519	0.428	0.665	0.521
		20	0.518	0.640	0.573	0.544	0.575	0.559	0.509	0.595	0.549
		40	0.584	0.644	**0.613**	**0.595**	0.612	0.603	0.519	0.639	0.573
Compact	synth	5	0.151	**0.696**	0.248	0.170	0.617	0.267	0.336	0.427	0.376
		10	0.176	0.663	0.278	0.195	0.624	0.297	0.295	0.486	0.367
		20	0.196	0.672	0.303	0.205	0.635	0.310	0.278	0.552	0.370
		40	0.166	0.643	0.264	0.197	0.595	0.296	0.306	0.530	0.388
	synth + human	5	0.600	0.641	0.620	0.552	0.635	0.591	0.588	0.600	0.594
		10	0.583	0.662	0.620	0.629	0.606	0.617	**0.656**	0.606	0.630
		20	0.622	0.647	0.634	0.597	0.688	0.639	0.600	0.660	0.629
		40	0.614	0.690	**0.650***	0.609	0.630	0.619	0.611	0.630	0.620

Table 2: Recall (R), precision (P) and F1 scores on development data. c indicates the number of sentences in the context provided to the tagger, EMR stands for entity modification rate. Best recall, precision and F1 scores for both generators (simple and compact) are shown in bold, the submitted model is identified by an asterisk (*).

from the shared task (which contains all error types), (2) **synthetic data** created by perturbing the human-written summaries from Rotowire (which contains only *NAME* and *NUMBER* errors; see Section 2.4 for details).

2.4 Synthetic Data

The gold-standard data contains only 60 games, as opposed to 3,395 games in the Rotowire training set. This led us to an idea of using the training set as a source of synthetic data for our model.

We create the synthetic data by introducing errors into human-written descriptions. We focus only on the *NAME* and *NUMBER* errors—the categories which are the most represented and also easiest to generate. In each sentence, we identify named entities in the text using *spaCy*.[5] We modify only certain portion of entities according to the *entity modification rate*, which we treat as a hyperparameter. We introduce the *NAME* errors by:

(1) swapping the names of teams with opponent teams,
(2) swapping the names of players with other players in the game,
(3) swapping the names of cities with other cities in the Rotowire dataset,
(4) modifying the days of the week.

For *NUMBER* errors, we take an integer n identified in the text, sample a number from a normal distribution with $\mu = n$ and $\sigma = 3$, and truncate it to get the integer. We re-sample if the output equals the original number, or for negative outputs. If the number is spelled out, we use `text2num`[6] and `num2words`[7] to convert to digits and back.

3 Experiments

We train a PyTorch version of RoBERTa from the Huggingface Transfomers repository (Wolf et al., 2019) using the AdamW optimizer (Loshchilov and Hutter, 2019), learning rate 5×10^{-5} and linear warmup. We finetune the model for 10 epochs and select the model with the highest validation score. We experiment with several hyperparameters:

(a) *simple* vs. *compact* sentences in G,
(b) *number of sentences* retrieved for the context: c = 5, 10, 20 or 40;
(c) *entity modification rate* (EMR): proportion of entities which are modified in the synthetic data: 0.25, 0.5, or 0.75.

We evaluate the model using a script provided by the organizers, which computes recall and precision of the model output with respect to the human-annotated data. Since we use the human-annotated data for training, we perform 6-fold cross-validation: in each run, we use 45 games for training, 5 games for validation, and 10 games for evaluation.

The results of our model on the development data are listed in Table 2.[8] For our final submission,

[5] https://spacy.io

[6] https://pypi.org/project/text2num/

[7] https://pypi.org/project/num2words/

[8] Due to space constraints, we do not list the results of

Error Type	Mistake		Token	
	R	P	R	P
NAME	0.750	0.846	0.759	0.862
NUMBER	0.777	0.750	0.759	0.752
WORD	0.514	0.483	0.465	0.529
CONTEXT	0.000	-	0.000	-
NOT_CHECKABLE	0.000	-	0.000	-
OTHER	0.000	-	0.000	-
Overall	0.691	0.756	0.550	0.769

Table 3: Results of our system on test data: recall (R) and precision (P) are shown for individual error types.

we selected the model with the best F1-score overall, which is 0.65 (61% recall and 69% precision). The model uses 40 compact sentences in context, 0.25 EMR and was trained on both synthetic and human-annotated data. However, note that the hyperparameters of the best models are quite varied. Although compact texts are generally helpful, there are also some well-performing models using simple templates only. A higher number of sentences in context may help to achieve better F1-score, but not always (the longer context is also sometimes cropped to fit the input). Using a higher EMR then generally leads to higher recall, suggesting that the model adapts to the base rate of errors.

4 Results of our Charles-UPF submission

Table 3 shows the results of our model on the official test data of the task, broken down by error types. The overall scores are higher than on the development set – test set recall is 0.691 (vs. 0.614 on the development set) and precision is 0.756 (vs. 0.690). The fact that we used the whole available human annotated data for training the final model may have contributed to the difference, but it is also possible that the test data was somewhat less challenging. We note that our model was able to identify only three types of errors (*NAME*, *NUMBER*, *WORD*), having better results for the *NAME* and *NUMBER* errors. We believe the explanation is two-fold: the names and numbers are often found verbatim in the input data (and in our generated facts), which makes them easy to detect, and also the corresponding error types were the most represented in the training data. In contrast, the three error types which were not detected are much less represented in the training data and hard to detect in our setup.

model trained only on annotated data. The results were overall in the 0.3-0.5 range for both recall and precision, and no model was the best-performing one in terms of any metric.

5 Discussion

Our Charles-UPF submission achieved the best results in the automatic metrics category, but there is still a gap with what humans can achieve, as shown by the Laval University submission's (Garneau and Lamontagne, 2021) overall 0.841 recall and 0.879 precision. One way to improve our system would be to enrich the reference fact descriptions, by either inferring more information from the raw data, or by extracting additional data from external databases.[2] Another option would be to add surrounding sentences to the context – this could help to resolve coreferences (e.g., if a player is referred to as *"He"*) and to detect the *CONTEXT* errors.

We also note that our approach requires the real system outputs manually annotated with errors in order to work well – using only synthetic data results in low recall (see Table 2). However, we believe that more sophisticated techniques for creating the synthetic data could help to achieve same results with less human-annotated data. We also believe that our system is in general applicable to new games or seasons. The rule-based generator does not need any adapting, the vocabulary of both neural parts (context selector and error tagger) is based on subwords and thus also able to handle unseen player/team/city names. The model effectively learns to compare entities from the context and the evaluated sentence, the absolute values are thus less important than their agreements and differences.

6 Conclusion

We presented our system for detecting errors in generated descriptions of basketball matches. Our system can automatically classify the errors on token level, using a pretrained language model and textual description of data generated by a rule-based NLG system. Our system reached 0.691 recall and 0.756 precision on the test data, finishing first out of four automatic metric submissions in the INLG 2021 Accuracy Evaluation shared task.

Acknowledgements

This work was supported by the Charles University projects GAUK 140320, SVV 260575, and PRIMUS/19/SCI/10, an Apple NLU Research Grant for Heriot-Watt University and Charles University, and by the European Commission via UPF under the H2020 program contract numbers 786731, 825079, 870930 and 952133.

References

Anya Belz, Simon Mille, and David M. Howcroft. 2020. Disentangling the properties of human evaluation methods: A classification system to support comparability, meta-evaluation and reproducibility testing. In *Proceedings of the 13th International Conference on Natural Language Generation*, pages 183–194, Dublin, Ireland. Association for Computational Linguistics.

Thiago Castro Ferreira, Claire Gardent, Nikolai Ilinykh, Chris van der Lee, Simon Mille, Diego Moussallem, and Anastasia Shimorina. 2020. 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+)*, pages 55–76, Dublin, Ireland (Virtual).

Bhuwan Dhingra, Manaal Faruqui, Ankur Parikh, Ming-Wei Chang, Dipanjan Das, and William Cohen. 2019. Handling divergent reference texts when evaluating table-to-text generation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4884–4895, Florence, Italy.

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 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*, pages 131–137, Dublin, Ireland. 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. ArXiv: 1901.07931.

Katja Filippova. 2020. Controlled hallucinations: Learning to generate faithfully from noisy data. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 864–870.

Nicolas Garneau and Luc Lamontagne. 2021. Shared task in evaluating accuracy: Leveraging pre-annotations in the validation process. In *Proceedings of the 14th International Conference on Natural Language Generation*, Aberdeen, Scotland, UK.

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. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 2410–2424.

Mihir Kale and Abhinav Rastogi. 2020. Text-to-text pre-training for data-to-text tasks. In *Proceedings of the 13th International Conference on Natural Language Generation*, pages 97–102.

Youngwoo Kim, Myungha Jang, and James Allan. 2020. Explaining text matching on neural natural language inference. *ACM Transactions on Information Systems (TOIS)*, 38(4):1–23.

Guy Lapalme. 2020. RDFjsRealB: a symbolic approach for generating text from RDF triples. In *Proceedings of the 3rd International Workshop on Natural Language Generation from the Semantic Web (WebNLG+)*, pages 144–153, Dublin, Ireland (Virtual).

Alon Lavie and Abhaya Agarwal. 2007. Meteor: An Automatic Metric for MT Evaluation with High Levels of Correlation with Human Judgments. In *Proceedings of the Second Workshop on Statistical Machine Translation*, pages 228–231.

Chris van der Lee, Albert Gatt, Emiel van Miltenburg, Sander Wubben, and Emiel Krahmer. 2019. Best practices for the human evaluation of automatically generated text. In *Proceedings of the 12th International Conference on Natural Language Generation*, pages 355–368, Tokyo, Japan. Association for Computational Linguistics.

Peiguang Li, Hongfeng Yu, Wenkai Zhang, Guangluan Xu, and Xian Sun. 2020. SA-NLI: A supervised attention based framework for natural language inference. *Neurocomputing*, 407:72–82.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A Robustly Optimized BERT Pretraining Approach. *arXiv preprint arXiv:1907.11692*.

Ilya Loshchilov and Frank Hutter. 2019. Fixing weight decay regularization in Adam. In *International Conference on Learning Representations (ICLR)*, New Orleans, LA, USA.

Simon Mille, Stamatia Dasiopoulou, Beatriz Fisas, and Leo Wanner. 2019. Teaching FORGe to verbalize DBpedia properties in Spanish. In *Proceedings of the 12th International Conference on Natural Language Generation*, pages 473–483, Tokyo, Japan. Association for Computational Linguistics.

Jekaterina Novikova, Ondřej Dušek, Amanda Cercas Curry, and Verena Rieser. 2017. Why we need new evaluation metrics for NLG. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2241–2252, Copenhagen, Denmark.

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, PA, USA.

Clément Rebuffel, Marco Roberti, Laure Soulier, Geoffrey Scoutheeten, Rossella Cancelliere, and Patrick Gallinari. 2021. Controlling hallucinations at word level in data-to-text generation. *arXiv preprint arXiv:2102.02810*.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*.

Ehud Reiter and Craig Thomson. 2020. Shared task on evaluating accuracy. In *Proceedings of the 13th International Conference on Natural Language Generation*, pages 227–231, Dublin, Ireland (Virtual).

Sashank Santhanam and Samira Shaikh. 2019. Towards best experiment design for evaluating dialogue system output. In *Proceedings of the 12th International Conference on Natural Language Generation*, pages 88–94, Tokyo, Japan. Association for Computational Linguistics.

Thibault Sellam, Dipanjan Das, and Ankur Parikh. 2020. BLEURT: Learning robust metrics for text generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7881–7892, Online.

Craig Thomson and Ehud Reiter. 2020a. A gold standard methodology for evaluating accuracy in data-to-text systems. In *Proceedings of the 13th International Conference on Natural Language Generation*, pages 158–168, Dublin, Ireland. Association for Computational Linguistics.

Craig Thomson and Ehud Reiter. 2020b. A gold standard methodology for evaluating accuracy in data-to-text systems. In *Proceedings of the 13th International Conference on Natural Language Generation*, pages 158–168, Dublin, Ireland (Virtual).

Craig Thomson and Ehud Reiter. 2021. Generation challenges: Results of the Accuracy Evaluation Shared Task. In *Proceedings of the 14th International Conference on Natural Language Generation*, Aberdeen, Scotland, UK.

James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. 2019. Generating token-level explanations for natural language inference. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, page 963–969, Minneapolis, MN, USA.

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*.

Trung Tran and Dang Tuan Nguyen. 2020. WebNLG 2020 challenge: Semantic template mining for generating references from RDF. In *Proceedings of the 3rd International Workshop on Natural Language Generation from the Semantic Web (WebNLG+)*, pages 177–185, Dublin, Ireland (Virtual).

Tsung-Hsien Wen, Milica Gašić, Nikola Mrkšić, Pei-Hao Su, David Vandyke, and Steve Young. 2015. Semantically conditioned LSTM-based natural language generation for spoken dialogue systems. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1711–1721, Lisbon, Portugal. Association for Computational Linguistics.

Sam Wiseman, Stuart M Shieber, and Alexander M 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.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2019. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*.

Shehel Yoosuf and Yin Yang. 2019. Fine-grained propaganda detection with fine-tuned BERT. In *Proceedings of the Second Workshop on Natural Language Processing for Internet Freedom: Censorship, Disinformation, and Propaganda*, pages 87–91, Hong Kong.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi. 2020. BERTScore: Evaluating text generation with BERT. In *International Conference on Learning Representations (ICLR)*, Online.

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*, pages 2481–2491, Online.