NUIG-DSI at the WebNLG+ challenge: Leveraging Transfer Learning for RDF-to-text generation

Nivranshu Pasricha[1], **Mihael Arcan**[2] and **Paul Buitelaar**[1,2]
[1]SFI Centre for Research and Training in Artificial Intelligence
[2] Insight SFI Research Centre for Data Analytics
Data Science Institute, National University of Ireland Galway
n.pasricha1@nuigalway.ie

Abstract

This paper describes the system submitted by NUIG-DSI to the WebNLG+ challenge 2020 in the RDF-to-text generation task for the English language. For this challenge, we leverage transfer learning by adopting the T5 model architecture for our submission and fine-tune the model on the WebNLG+ corpus. Our submission ranks among the top five systems for most of the automatic evaluation metrics achieving a BLEU score of 51.74 over *all* categories with scores of 58.23 and 45.57 across *seen* and *unseen* categories respectively.

1 Introduction

The WebNLG+ challenge (Castro-Ferreira et al., 2020) is concerned with mapping data to text, where the data is in the form of RDF-triples[1] extracted from DBpedia (Auer et al., 2007) and the text is a verbalisation of these triples. The RDF-to-text generation task focuses on generating a verbalisation in a human language in the output based on a set of RDF-triples in the input. In general, data-to-text generation is concerned with building systems that can produce meaningful texts in a natural language from some underlying non-linguistic representation of information (Reiter and Dale, 2000). Traditionally, most applications for data-to-text generation have relied on rule-based systems which are designed using a modular pipeline architecture (Gatt and Krahmer, 2018). However, there has been a shift recently towards end-to-end architectures using neural networks to convert data in the input to text in a natural language in the output. In our submission, we employ an end-to-end approach using the T5 model architecture (Raffel et al., 2020) which is pre-trained on a large corpus of text scraped from the Web. We fine-tune the T5 model on the WebNLG+ corpus and explore various pre-training and pre-processing strategies to improve the performance of our system.

[1]RDF - Resource Description Framework

2 Background

The WebNLG challenge (Gardent et al., 2017) was created with the goal of producing a common benchmark to compare "microplanners", i.e, generation systems that verbalise non-linguistic content to text in some human language. In 2017, the challenge received a mix of submissions built using template or grammar-based pipeline, statistical machine translation (SMT) and neural machine translation (NMT) frameworks. The test set used for final evaluation was split into two subsets, *seen* and *unseen*. The first subset contained data from the categories that were also present in the training set while the second included new data from unseen categories that were not present in the training set at all. On the *seen* categories, the NMT and SMT-based systems mostly outperformed the rule-based pipeline sytems in terms of BLEU and TER score. However, the scores for the NMT-based systems dropped significantly on the *unseen* categories while the rule-based systems were able to generalise better on the new and unseen domains.

Further work by Castro Ferreira et al. (2019) compared pipeline-based and end-to-end architectures and their findings also suggest that the systems which are trained end-to-end are comparable to pipeline methods on the *seen* categories but do not generalise to new and *unseen* domains of data. More recently, Kale (2020) have shown that applying transfer learning using an end-to-end pretrained model such as T5 achieves state-of-the-art results on three benchmark datasets for data-to-text generation and performs well even on out-of-domain inputs in the *unseen* categories of data.

The T5 model (Raffel et al., 2020) follows a transformer-based encoder-decoder architecture

3rd International Workshop on Natural Language Generation from the Semantic Web (WebNLG+), Dublin, Ireland (Virtual), 18 December 2020, pages 137–143, ©*2020 Association for Computational Linguistics*

Tripleset	200 Public Square Cleveland Cleveland	location leader isPartOf	Cleveland Frank G. Jackson Ohio
T5-*prefix*	*Translate triple to text:* 200 Public Square location Cleveland Cleveland leader Frank G. Jackson Cleveland isPartOf Ohio		
tags	<*SUB*> 200 Public Square <*PRED*> location <*OBJ*> Cleveland <*SUB*> Cleveland <*PRED*> leader <*OBJ*> Frank G. Jackson <*SUB*> Cleveland <*PRED*> isPartOf <*OBJ*> Ohio		
types	<*BUILDING*> 200 Public Square <*PRED*> location <*CITY*> Cleveland <*CITY*> Cleveland <*PRED*> leader <*POLITICIAN*> Frank G. Jackson <*CITY*> Cleveland <*PRED*> isPartOf <*POPULATEDPLACE*> Ohio		
split-predicate	200 Public Square *location* Cleveland Cleveland *leader* Frank G. Jackson Cleveland *is part of* Ohio		
Lexicalisation	Frank G Jackson is a leader in Cleveland, Ohio where 200 Public Square is located.		

Figure 1: Example of a tripleset (top) with different pre-processing strategies for linearisation and fine-tuning (middle) and reference lexicalisation (bottom).

(Vaswani et al., 2017) and is pre-trained using unsupervised learning on a large corpus of unlabeled data obtained from the Web using the Common Crawl project. It is trained using a denoising objective, also known as "masked language modelling", where a model is trained to predict missing or otherwise corrupted tokens in the input. The basic idea underlying the T5 model is to treat every text processing problem as a "text-to-text" problem, i.e. taking some form of text in the input and producing new text as the output. For the WebNLG+ dataset, this input is not purely textual in the form of sentences but rather a set of RDF-triples. Therefore, for modelling the input with the T5 architecture, we linearise the triples in the input into a sequence and effectively treat it as a linear sequence of text.

3 Dataset

The WebNLG+ dataset consists of RDF-triples extracted from DBpedia paired with reference text lexicalisations. These lexicalisations contain sequences of one or more short sentences in English and Russian, verbalising the data units in the input. The corpus contains RDF-triples from 19 DBpedia categories and is divided into two subsets, *seen* and *unseen*. The 16 *seen* categories are *Airport*, *Artist*, *Astronaut*, *Athlete*, *Building*, *CelestialBody*, *City*, *ComicsCharacter*, *Company*, *Food*, *MeanOfTransportation*, *Monument*, *Politician*, *SportsTeam*, *University* and *WrittenWork* and the three *unseen* categories are *Film*, *MusicalWork* and *Scientist*. The English corpus contains 16,095 RDF-triples in the input paired with 42,873 lexicalisations in the output. We use the same training, validation and test sets as defined in the challenge, where the training set contains data only from the *seen* categories and the test set contains data from both *seen* and *unseen* categories (Table 1).

Number of ...	train	dev	test
... data-text pairs	35,426	4,464	–
... triplesets	13,211	1,667	1,779
... triples	38,399	4,841	5,639
... entities	3,729	2,375	832
... predicates	372	290	220
... lexicalisation tokens	702,482	88,428	–
... lexicalisation types	14,805	7,010	–

Table 1: Statistics for the English WebNLG+ dataset.

4 Methodology

The T5 model is pre-trained on a multi-task mixture of supervised and unsupervised tasks where each task is converted into a text-to-text format. This model can then be fine-tuned on a downstream supervised task on some labeled data or further trained in an unsupervised fashion on unlabeled data. In this section, we explore various pre-processing strategies for fine-tuning as well as a pre-training strategy for the T5 model.

Fine-tuning: For fine-tuning the T5 model, we linearise the triples in the input into a sequence and prepend it with the string, *Translate triple to text:* to all instances in the dataset, similar to the implementation of the original T5 model. We follow the default ordering for the triples when linearising a tripleset. We also incorporate additional tags to mark the subject, predicate and object in each triple using <*SUB*>, <*PRED*> and <*OBJ*> tags respectively. These tags are added as additional special tokens to the model vocabulary. Furthermore, for the subject and the object entities in each triple, we add tags for the type of the entity using DBpe-

	All	Seen	Unseen	All	Seen	Unseen	All	Seen	Unseen
		BLEU (↑)			**METEOR** (↑)			**TER** (↓)	
Baseline									
LSTM	35.0	53.8	4.3	26.3	39.0	8.2	72.6	53.6	97.5
Transformer	36.8	54.6	3.7	27.2	40.7	7.4	68.6	53.1	89.3
Fine-tuning T5									
T5	53.4	62.8	37.4	37.4	40.4	36.0	52.7	45.1	64.6
T5 + *tags*	53.5	61.7	39.5	37.1	39.9	36.5	52.8	46.1	63.3
T5 + *types*	50.3	57.6	38.9	34.1	36.1	34.5	54.7	49.6	62.6
T5 + *split-predicate*	53.4	62.4	38.1	37.4	40.5	36.1	52.6	45.2	64.3
Additional Pre-training + Fine-tuning T5									
T5 + *lex*	54.1	64.5	37.1	37.9	41.2	36.3	52.0	43.1	65.9
T5 + *abstracts*	54.8	64.6	38.6	38.1	41.4	36.5	51.0	42.9	63.5

Table 2: Results for *all*, *seen* and *unseen* categories in the validation (dev) set for baseline and T5-small models.

dia. In the cases where we cannot assign the type to an entity, we check whether it is a date or a numerical value and assign it the type <*TIMEPERIOD*> and <*NUMERIC*> respectively. Otherwise, the type is taken to be <*UNKNOWN*>.

In many instances, the predicate in the RDF-triples is made up of two or more tokens joined together using the *camelCase* convention. We split these multi-word predicates on the *camelCase* into separate tokens using a regular expression. Figure 1 shows an example of a tripleset paired with the modifications mentioned in this section along with the corresponding lexicalisation. The T5 model is then trained in a supervised fashion with the RDF-triples in the input to generate the target lexicalisations in the output.

Pre-training: The T5 model can additionally be trained on unlabeled data with masked spans of tokens with the objective to predict the missing tokens. In our experiments, we pre-train two models using this strategy on two different corpora. In the first case, we train on the reference lexicalisations in the WebNLG+ corpus by randomly corrupting 15% of the tokens in the text and for the second we use a corpus of abstracts from DBpedia. We include abstracts only for the entities which are present in the training set and randomly mask 15% of the tokens in this case too. Since we cannot find an abstract for each and every entity in the training set, we ended up with 2,540 abstracts consisting of 398,864 tokens and 50,092 types in total with an average of 157.03 tokens per abstract. After pre-training, we fine-tune on the WebNLG+ corpus to predict the target lexicalisation in the output conditioned on the RDF-triples given in the input.

5 Experimental Setup

We adopt the WebNLG baseline system (Gardent et al., 2017) as one of the baseline architectures for our experiments, which is a vanilla sequence-to-sequence LSTM model with attention (Bahdanau et al., 2015) where the RDF-triples in the input are linearised as a sequence and the output text is tokenised before training. We use another baseline based on the transformer architecture (Vaswani et al., 2017) similar to the end-to-end architecture setup by Castro Ferreira et al. (2019).

These baseline models are trained using the OpenNMT library (Klein et al., 2017). We use the default parameters for two baseline models. Two hidden layers and 500 units per hidden layer with input feeding (Luong et al., 2015) enabled and word embeddings of size 500-dimensions are used for the LSTM neural model. Dropout is applied with value 0.3 and the LSTM model is trained with stochastic gradient descent, starting with a learning rate of 1.0 and learning rate decay enabled. For the transformer model, the encoder-decoder setup contains 6 layers with 512 hidden units. The word embeddings are 512-dimensional and the feed-forward sublayers are 2048-dimensional. Each multi-head attention sublayer consists of 8 attention heads. Dropout is applied with value 0.1 and the model is trained using Adam optimizer (Kingma and Ba, 2015) for 100,000 steps. We also enable the options for dynamic dictionary and shared vocabulary to allow the model to share tokens between the source and the target side.

For the pre-trained T5 model, we follow HuggingFace's (Wolf et al., 2019) implementation of the T5 architecture to train our systems. We adopt

	all	seen	unseen entities	unseen categories
BLEU	51.74	58.26	52.76	45.57
BLEU_NLTK	0.514	0.579	0.523	0.454
METEOR	0.403	0.416	0.415	0.388
CHRF++	0.669	0.699	0.691	0.632
TER	0.417	0.408	0.381	0.438
BERT_PREC	0.959	0.964	0.964	0.953
BERT_REC	0.954	0.958	0.961	0.949
BERT_F1	0.956	0.960	0.962	0.950
BLEURT	0.61	0.60	0.67	0.58

Table 3: Results from automatic evaluation on the WebNLG+ test set, following the *tags* + *split-predicate* strategy for the "base" variant of the T5-model.

the "small" variant of the T5 architecture for fine-tuning and additional pre-training in our experiments. The T5-small model contains 6 layers each in the encoder and the decoder with each multi-head attention sublayer consisting of 8 heads. The word-embeddings are 512-dimensional and the feed-forward sublayers are 2048-dimensional. This variant has about 60 million parameters and is faster to train compared to other variants of the T5 architecture. The T5-small variant closely follows the architectural set-up of the baseline transformer model as the two models are roughly equivalent in terms of the structure of the encoder and decoder layers, and the number of parameters. For our final submission, however, we use the "base" variant which consists of 12 layers each in the encoder and the decoder with each multi-head attention sublayer consisting of 12 heads. In this variation, the word-embeddings are 768-dimensional and the dimensionality of the feed-forward sublayers is 3072. This variant has about 220 million parameters. In both variants of the T5 model architecture, weight decay is applied with a value of 0.01 and dropout is applied with a probability of 0.1 for regularisation. Fine-tuning as well as additional pre-training is done for 20,000 steps respectively with a batch-size of 32 using the Adam optimizer with a learning rate of 0.001.

6 Results

In this section, we report the results of our experiments on the validation set of the WebNLG+ corpus. Since at the time of writing, we do not have access to the official WebNLG+ reference lexicalisations in the test set, to evaluate performance on the *unseen* categories of data, we treat *Artist*, *Athlete*, *CelestialBody*, *Company*, *MeanOfTransportation* and *Politician* as *unseen* categories and exclude data from these categories from the training set to treat them as new and *unseen* categories in the validation set.

Table 2 shows the results of automatic evaluation in terms of three commonly used evaluation metrics, BLEU (Papineni et al., 2002), METEOR (Denkowski and Lavie, 2014) and TER (Snover et al., 2006). The LSTM and transformer baseline models achieve a BLEU score of 35.0 and 36.8 respectively across *all* categories of data and a score of about 54 on the *seen* categories. However, there is a significant drop in the performance on the *unseen* categories, which shows that end-to-end trained systems do not generalise well to new and unseen domains of data. The results from fine-tuning the T5 model in Table 2 indicate that transfer learning is hugely beneficial in the context of RDF-to-text generation as it achieves significant gains over the baselines right out of the box. Even though the baseline transformer model and the T5-small model have roughly equivalent architecture set-up, the T5 model performs much better across all categories of data. For the *seen* categories, it shows an improvement of about 8 – 9 points in terms of the BLEU and TER metrics, while for the *unseen* categories, the improvement is by more than 30 points over the baseline models. The gains in performance here can be attributed to the fact that the T5 model is pre-trained on a large unlabeled corpus of data, while the baseline transformer model is trained from scratch.

In terms of METEOR score, the gains of transfer learning are noticeable only in the *unseen* categories of data, while for the data in the *seen* categories, the baseline models are quite competitive and there does not appear to be any significant improvements with the pre-trained model.

The addition of *<SUB>*, *<PRED>* and *<OBJ>* tags in the T5+*tags* model improves the BLEU score for *unseen* categories by more than 2 points from 37.4 to 39.5. However, for *seen* categories, there is a drop of about of 0.9. Information about entity types from DBpedia also appears to be useful for the *unseen* categories, improving the BLEU score from 37.4 to 38.9 for the T5+*types* model. However, it also leads to a performance drop by about 4 points for each metric in the case of *seen* categories. The T5 model uses SentencePiece (Kudo and Richardson, 2018) for subword tokenisation to handle unknown and rare tokens, such as the multi-word predicates in this corpus. However,

	Data Coverage	Relevance	Correctness	Text Structure	Fluency
all categories					
Baseline 1	92.892 (0.17)	93.784 (0.161)	91.794 (0.19)	87.4 (0.039)	82.43 (0.011)
Baseline 2	92.066 (0.127)	92.588 (0.113)	90.138 (0.13)	85.737 (-0.064)	80.941 (-0.143)
NUIG-DSI	92.063 (0.116)	94.061 (0.161)	92.053 (0.189)	91.588 (0.258)	88.898 (0.233)
Reference	95.442 (0.251)	94.392 (0.139)	94.149 (0.256)	92.105 (0.254)	89.846 (0.279)
seen categories					
Baseline 1	95.296 (0.28)	94.568 (0.153)	93.593 (0.226)	87.04 (0.074)	82.664 (0.03)
Baseline 2	90.253 (0.065)	89.568 (-0.043)	87.608 (0.042)	82.892 (-0.16)	75.037 (-0.406)
NUIG-DSI	91.253 (0.059)	94.512 (0.178)	92.494 (0.162)	90.744 (0.234)	88.611 (0.18)
Reference	95.491 (0.264)	94.142 (0.135)	93.355 (0.236)	91.225 (0.198)	88.136 (0.225)
unseen categories					
Baseline 1	91.201 (0.106)	92.312 (0.12)	90.32 (0.163)	87.264 (0.039)	82.414 (-0.007)
Baseline 2	93.13 (0.137)	93.948 (0.145)	91.213 (0.105)	87.542 (-0.032)	83.473 (-0.057)
NUIG-DSI	92.697 (0.13)	93.937 (0.142)	91.613 (0.175)	91.494 (0.23)	88.787 (0.237)
Reference	95.178 (0.23)	93.389 (0.066)	94.207 (0.263)	92.19 (0.277)	90.508 (0.31)
unseen entities					
Baseline 1	93.36 (0.161)	96.099 (0.271)	92.635 (0.199)	88.243 (-0.011)	82.126 (0.025)
Baseline 2	92.207 (0.196)	93.797 (0.266)	91.302 (0.317)	85.644 (0.003)	83.604 (0.039)
NUIG-DSI	91.752 (0.165)	93.694 (0.181)	92.446 (0.26)	93.041 (0.358)	89.577 (0.303)
Reference	95.991 (0.283)	97.117 (0.315)	95.171 (0.268)	93.189 (0.281)	90.788 (0.285)

Table 4: Results from human evaluation (with normalized z-scores) on the WebNLG+ test set. Baseline 1 is a grammar-based system based on Mille et al. (2019), while Baseline 2 is based on the FORGe system (Mille and Dasiopoulou, 2017). Our submission follows the *tags + split-predicate* strategy for the "base" variant of the pre-trained T5 model.

we find that explicitly splitting the predicates on *camelCase* into constituent tokens appears to be helpful for *unseen* categories of data as shown for the T5+*split-predicate* model in Table 2.

For the small variant of the T5 model architecture, additional unsupervised pre-training on reference lexicalisations (T5+*lex* in Table 2) and abstracts (T5+*abstracts* in Table 2) from DBpedia appears to be useful for both *seen* and *unseen* categories of data. In this work, we included abstracts from DBpedia for only the entities that are present in the training set. Future work can explore other sources of unlabeled data combined with a pre-training strategy relevant for this task.

For our final submission to the WebNLG+ challenge 2020, we train a "base" variant of the T5 model using data from the entire training set of the WebNLG+ corpus. Before fine-tuning the T5-base model, we split the multi-word predicates and add *<SUB>*, *<PRED>* and *<OBJ>* tags for subjects, predicates and objects respectively. Table 3 shows the automatic evaluation results for our submission using the GERBIL NLG framework (Moussalem et al., 2020) on the WebNLG+ test set in terms of chrf++ (Popović, 2017), BERT score (Zhang et al., 2020) and BLEURT (Sellam et al., 2020) along with BLEU, METEOR and TER scores. Our system ranks among the top 5 for most of these evaluation metrics across all categories. In terms of BLEU score, our submission achieves scores of 58.26 for *seen* categories and 45.52 for the *unseen* categories. For the test set containing *unseen* entities, our system achieves the highest BLEU score of 52.76 and ranks among the top two for most of the automatic evaluation metrics.

Table 4 shows results of human evaluation on the WebNLG+ test set for our submission along with two baselines and the reference lexicalisation. For the evaluation, human annotators were asked to what extent they agree with the statements depicting five criteria of data coverage, relevance, correctness, text structure and fluency. For data coverage, our submission (NUIG-DSI) as well as both grammar-based baselines achieve a similar score across *all* categories. On the *seen* subset, our system performs better compared to Baseline 2, however, Baseline 1 achieves an even higher score. In terms or relevance and correctness, our end-to-end system based on the pre-trained "base" variant of the T5 model performs better than Baseline 2 while achieving similar scores and rank compared to Baseline 1. For the metrics measuring fluency and text structure, our submission achieves much higher scores than the baselines, by more than 5 points in some instances, which is not surprising since Wiseman et al. (2017) have also shown that neural end-to-end approaches for data-to-text generation are quite good at producing fluent outputs but can struggle to get the factual information in the input correctly in the output. Overall, our system achieves a rank of 2 for data coverage and a rank of 1 for the rest for the human evaluation metrics.

7 Conclusion

In this paper, we presented the description of the system submitted by NUIG-DSI to the WebNLG+ challenge 2020. We participated in the RDF-to-text generation task for the English language using an end-to-end system based on the T5 model architecture. We split the predicates on *camelCase* and add *<SUB>*, *<PRED>* and *<OBJ>* tags for subjects, predicates and objects respectively before fine-tuning the T5 model on the WebNLG+ corpus. Our submission ranks among the top 5 systems for most of the automatic evaluation metrics, achieving a BLEU score of 51.74 over *all* categories.

Acknowledgments

This work was conducted with the financial support of the Science Foundation Ireland Centre for Research Training in Artificial Intelligence under Grant No. 18/CRT/6223 and co-supported by Science Foundation Ireland under grant number SFI/12/RC/2289 2 (Insight), co-funded by the European Regional Development Fund.

References

Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary Ives. 2007. Dbpedia: A nucleus for a web of open data. In *The semantic web*, pages 722–735. Springer.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural Machine Translation by Jointly Learning to Align and Translate. *Proceedings of the Third International Conference on Learning Representations*.

Thiago Castro-Ferreira, Claire Gardent, Nikolai Ilinykh, Chris van der Lee, Simon Mille, Diego Moussalem, and Anastasia Shimorina. 2020. The 2020 bilingual, bi-directional webnlg+ shared task: Overview and evaluation results (webnlg+ 2020). In *Proceedings of the 3rd WebNLG Workshop on Natural Language Generation from the Semantic Web (WebNLG+ 2020)*, Dublin, Ireland (Virtual). 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, Hong Kong, China. Association for Computational Linguistics.

Michael Denkowski and Alon Lavie. 2014. Meteor universal: Language specific translation evaluation for any target language. In *Proceedings of the Ninth Workshop on Statistical Machine Translation*, pages 376–380, Baltimore, Maryland, USA. 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, Santiago de Compostela, Spain. 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.

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

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Guillaume Klein, Yoon Kim, Yuntian Deng, Jean Senellart, and Alexander Rush. 2017. OpenNMT: Open-source toolkit for neural machine translation. In *Proceedings of ACL 2017, System Demonstrations*, pages 67–72, Vancouver, Canada. Association for Computational Linguistics.

Taku Kudo and John Richardson. 2018. SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 66–71, Brussels, Belgium. Association for Computational Linguistics.

Minh-Thang Luong, Hieu Pham, and Christopher D Manning. 2015. Effective Approaches to Attention-based Neural Machine Translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1412–1421.

S. Mille and S. Dasiopoulou. 2017. Forge at webnlg 2017. Technical report, Dept. of Engineering and Information and Communication Technologies, Universitat Pompeu Fabra.

Simon Mille, Stamatia Dasiopoulou, and Leo Wanner. 2019. A portable grammar-based nlg system for verbalization of structured data. In *Proceedings of the 34th ACM/SIGAPP Symposium on Applied Computing*, SAC '19, page 1054–1056, New York, NY, USA. Association for Computing Machinery.

Diego Moussalem, Paramjot Kaur, Thiago Castro-Ferreira, Chris van der Lee, Conrads Felix Shimorina, Anastasia, Michael Röder, René Speck, Claire Gardent, Simon Mille, Nikolai Ilinykh, and Axel-Cyrille Ngonga Ngomo. 2020. A general benchmarking framework for text generation. In *Proceedings of the 3rd WebNLG Workshop on Natural Language Generation from the Semantic Web (WebNLG+ 2020)*, Dublin, Ireland (Virtual). 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.

Maja Popović. 2017. chrf++: words helping character n-grams. In *Proceedings of the second conference on machine translation*, pages 612–618.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Ehud Reiter and Robert Dale. 2000. *Building Natural Language Generation Systems*, 1 edition. Cambridge University Press.

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. Association for Computational Linguistics.

Matthew Snover, Bonnie Dorr, Richard Schwartz, Linnea Micciulla, and John Makhoul. 2006. A study of translation edit rate with targeted human annotation. In *In Proceedings of Association for Machine Translation in the Americas*, pages 223–231.

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

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, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2019. Huggingface's transformers: State-of-the-art natural language processing. *ArXiv*, abs/1910.03771.

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*.