Proceedings of the 13th Conference on Language Resources and Evaluation (LREC 2022), pages 3542–3551
Marseille, 20-25 June 2022

Generating Textual Explanations for Machine Learning Models Performance: A Table-to-Text Task

Isaac Ampomah[1], James Burton[1], Amir Enshaei[2,3], Noura Al Moubayed[1]
[1]Durham University, [2]Newcastle University, [3]Caspian Learning Ltd
{isaac.k.ampomah, james.burton, noura.al-moubayed}@durham.ac.uk, amir.enshaei@newcastle.ac.uk

Abstract

Numerical tables are widely employed to communicate or report the classification performance of machine learning (ML) models with respect to a set of evaluation metrics. For non-experts, domain knowledge is required to fully understand and interpret the information presented by numerical tables. This paper proposes a new natural language generation (NLG) task where neural models are trained to generate textual explanations, analytically describing the classification performance of ML models based on the metrics' scores reported in the tables. Presenting the generated texts along with the numerical tables will allow for a better understanding of the classification performance of ML models. We constructed a dataset comprising numerical tables paired with their corresponding textual explanations written by experts to facilitate this NLG task. Experiments on the dataset are conducted by fine-tuning pre-trained language models (T5 and BART) to generate analytical textual explanations conditioned on the information in the tables. Furthermore, we propose a neural module, *Metrics Processing Unit* (MPU), to improve the performance of the baselines in terms of correctly verbalising the information in the corresponding table. Evaluation and analysis conducted indicate, that exploring pre-trained models for data-to-text generation leads to better generalisation performance and can produce high-quality textual explanations.

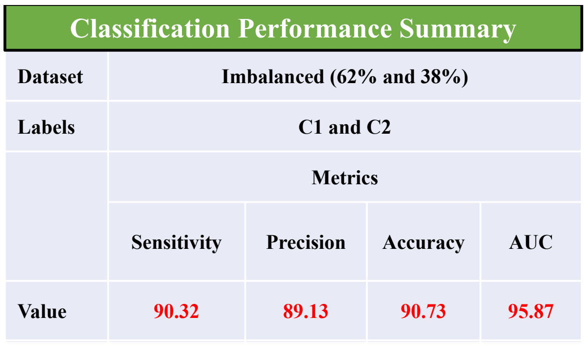
Keywords: Table-to-text generation, Structured data, Model performance summarisation

1. Introduction

Structured data-to-text generation is a natural language generation (NLG) task aiming to generate text from a structured source of data, most commonly either graphs or tables (Wen et al., 2015; Liu et al., 2018; Strauss and Kipp, 2008). Earlier work on data-to-text generation predominantly used rule-based methods (Goldberg et al., 1994; Reiter and Dale, 1997; Strauss and Kipp, 2008). These methods generate natural language text by employing linguistic rules and heuristics to select and populate pre-defined templates. However, a typical NLG system requires different sets of rules to perform content determination, text planning, sentence planning and surface realisation modules (Goldberg et al., 1994; van der Lee et al., 2017). This makes traditional NLG models difficult to maintain and less generalised.

Recently, leveraging deep neural methods for NLG has been shown to outperform existing rule-based methods (Wen et al., 2015; Liu et al., 2018; Puduppully et al., 2019; Parikh et al., 2020; Suadaa et al., 2021). These models are usually trained end-to-end without the need for pre-defined linguistic rules. In broader terms, transfer learning has been shown to produce close to state-of-the-art performance for downstream NLP tasks with a limited amount of the dataset by utilising large pre-trained language models (Devlin et al., 2019; Radford et al., 2019). Furthermore, (Peng et al., 2020; Chen et al., 2020c) argue that pre-trained language models (GPT-2 (Radford et al., 2019) and T5 (Raffel et al., 2020)) can indeed improve performance on structured data-to-text task.

The performance of trained ML models is widely reported using numerical tables (for example, see the table in Fig. 1) and graphs. However, background and



Classification Performance Summary				
Dataset	**Imbalanced (62% and 38%)**			
Labels	**C1 and C2**			
	Metrics			
	Sensitivity	**Precision**	**Accuracy**	**AUC**
Value	90.32	89.13	90.73	95.87

Figure 1: A table summarising the classification performance of a classifier.

domain knowledge are required to make sense of the graphs and tables. Therefore, non-experts will find it more challenging to fully understand the implications of the model's scores across the metrics. In response, this work presents a study on training neural models to generate textual explanations that analytically describe the classification performance of machine learning models. The generated textual explanation is based on the evaluation metrics' scores achieved, along with information on the underlying dataset (class labels and dataset distribution across the classes) of an arbitrary classification problem. The neural models are trained on table-explanation pairs annotated by computer science experts. Due to the limited size of our dataset, experiments are conducted by fine-tuning the pre-trained language models T5 (Raffel et al., 2020) and BART (Lewis et al., 2020). These pre-trained models treat all text-based language tasks as text-to-text generation; therefore, following (Moryossef et al., 2019; Chen et al., 2020b; Suadaa et al., 2021), the performance summary tables are linearised as flat strings. However, converting structured data to flat strings can result in the loss of important information and relations (Mager et al., 2020; Hoyle et al., 2021; Suadaa et al., 2021), therefore, exploring strategies to improve the encoding of the structured data can further improve the quality of the output generated texts. To this end, we propose a neural module, *Metrics Processing Unit* (MPU), to improve the performance of the pre-trained language models in terms of producing textual explanations verbalising correctly the information in the corresponding table. The MPU is employed to learn a semantic representation by directly encoding the information about the metrics from the table. The encoder combines the MPU's output representation with the embedding of the linearised representation to generate the contextualised joint representation information passed to the decoder. The contributions of this work are as follows:

- Introducing a new dataset[1] for generating analytical textual explanations describing the performance of classifiers on several machine learning tasks. The textual explanations are written by computer science experts and checked manually to ensure that they accurately reflect or verbalise the information in the corresponding performance table. To the best of our knowledge, this is the first of its kind to focus on explaining the performance of ML models.
- Proposing a neural module, *Metrics Processing Unit* (MPU), which improves the encoding of the information about the metrics ensuring that the outputs of the pre-trained models accurately verbalise the performance report summarised by the related table.
- Experiments with state-of-the-art neural models demonstrating the opportunities and challenges for future research on this table-to-text generation task.

The remainder of the paper is organised as follows: Section 2 briefly reviews the related works and Section 3 introduces the model performance explanations dataset. Section 4 introduces the models trained on the proposed dataset, the experiments conducted are presented in Section 5, and the results are compared and discussed in Section 6. Finally, the conclusions are presented in Section 7, together with avenues for future work.

2. Related Works

2.1. Natural Language Generation (NLG)

Generating text from structured data has been a persistent NLG task over the years (Kukich, 1983; Reiter and Dale, 1997; Goldberg et al., 1994). Earlier work on data-to-text generation predominately employed rule-based methods (Goldberg et al., 1994; McKeown, 1992; Reiter and Dale, 1997; Strauss and Kipp, 2008). These methods generate natural language text by employing linguistic rules and heuristics to select and populate pre-defined templates. A typical NLG system requires different sets of rules to perform the content determination and text planning, sentence planning and surface realisation modules (Goldberg et al., 1994; McKeown, 1992; van der Lee et al., 2017). This makes traditional NLG models difficult to maintain and less generalisable. Furthermore, the output texts lack flexibility and diversity given that the same set of templates are used for text generations.

Recent NLG research studies are moving towards exploring deep neural applications to automatically generate texts from structured data without relying on hand-engineered features and rules. These applications are table-to-text (Liu et al., 2018; Parikh et al., 2020; Su et al., 2021; Suadaa et al., 2021), table-based question answering (Wang et al., 2018; Chen et al., 2020a; Chemmengath et al., 2021), and graph to text generation (Ribeiro et al., 2020; Koncel-Kedziorski et al., 2019). However, neural models, despite their appeal, are data-hungry models requiring a large amount of high quality data to achieve higher generation performance.

The findings of the research works (Su et al., 2021; Peng et al., 2020; Chen et al., 2020b) demonstrate that state-of-the-art pre-trained language models such as GPT-2 (Radford et al., 2019), T5 (Raffel et al., 2020) and BART (Lewis et al., 2020) can be exploited to improve the performance on text generation tasks with limited amount of dataset. These pre-trained language models are well suited for text-to-text; hence applications to data-to-text tasks such as table-to-text and graphs-to-text require converting the structured data into linearised strings (Moryossef et al., 2019; Chen et al., 2020b; Hoyle et al., 2021). Despite this solution, the findings of (Mager et al., 2020; Hoyle et al., 2021; Suadaa et al., 2021) argue that the linearisation process can result in the loss of important information and relations. Therefore, in this work we propose the *Metrics Processing Unit* (MPU), which improves the encoding of the information about the metrics ensuring that the generated texts from the pre-trained models accurately verbalise the performance report summarised by the related table.

2.2. Dataset and Task Design

Existing datasets for table-to-text generation task include ROBOCUP (Chen and Mooney, 2008), ROTOWIRE (Wiseman et al., 2017), E2E (Novikova et al., 2016; Novikova et al., 2017), KBGEN (Banik et al., 2013), WEATHERGOV (Liang et al., 2009), and WIKIBIO (Lebret et al., 2016). These datasets cover different topics and themes. For example, E2E focuses on

[1]https://github.com/Durham-University-VIVID-Noura-s-Lab/ClassificationPerformanceExplanations

Property	Original	Permuted
size: train/test	725 / 100	4529 / 100
Unique words	3548	3548
Min. / Max. summary length	35/160	
# summaries with 2 sentences	113	579
# summaries with 3 sentences	347	1844
# summaries with >3 sentences	368	2206

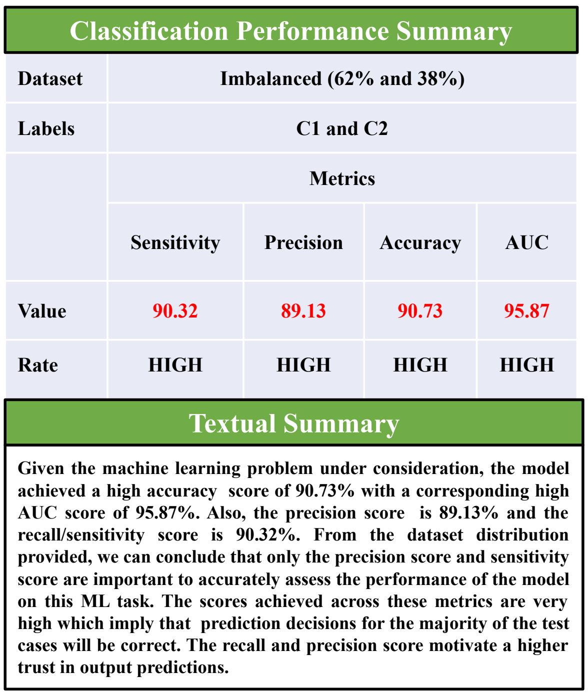
Table 1: Statistics of the model performance narrations dataset

generating restaurant descriptions, ROTOWIRE generates summaries of basketball sporting event and WIKIBIO focuses on producing biographies from a given Wikipedia bio-table.
Numerical tables are usually employed to report the performance of trained ML models. Understanding these numerical tables requires background and domain knowledge; hence augmenting the numerical tables with analytical texts will enable non-experts to find it easy to make sense of the scores achieved by the model on the arbitrary ML task. To the best of our knowledge, there is no work conducted on generating textual explanations describing the performance of ML models trained on arbitrary prediction tasks. The dataset most similar to the one introduced in this paper is the NUMERICAL TABLE-TO-TEXT proposed by Suadaa et al. (2021). Unlike (Suadaa et al., 2021), our dataset is not curated from numerical tables of experimental results published in scientific research articles. We trained different ML models on 59 classification tasks, and the performance of each model was summarised in numerical tables. For each table, computer science experts were tasked to provide analytical statements describing the performance of the model on the corresponding classification task. Fig. 2 shows an example of a classification performance summary table and the corresponding textual explanation.

3. Performance Explanations Dataset

To acquire the dataset for this study, different ML models were trained on 59 classification tasks across different application domains. Across each classification task, five different classifiers were trained. These classification models include random forest, support vector machines, logistic regression and K-nearest neighbour (KNN). For simplicity, only the common classification metrics (accuracy, precision, AUC, recall, specificity, F1-score, and F2-score) were considered. Ten computer science experts were hired to provide the analytical textual explanations. For each classifier, the expert is tasked to provide sentences summarising the performance based on the information in the corresponding table. To guide the annotators, they were instructed to respond (in English) to the following:

a Provide a summary of the scores achieved by the model across the evaluation metrics.



Classification Performance Summary

Dataset	Imbalanced (62% and 38%)			
Labels	C1 and C2			
	Metrics			
	Sensitivity	Precision	Accuracy	AUC
Value	90.32	89.13	90.73	95.87
Rate	HIGH	HIGH	HIGH	HIGH

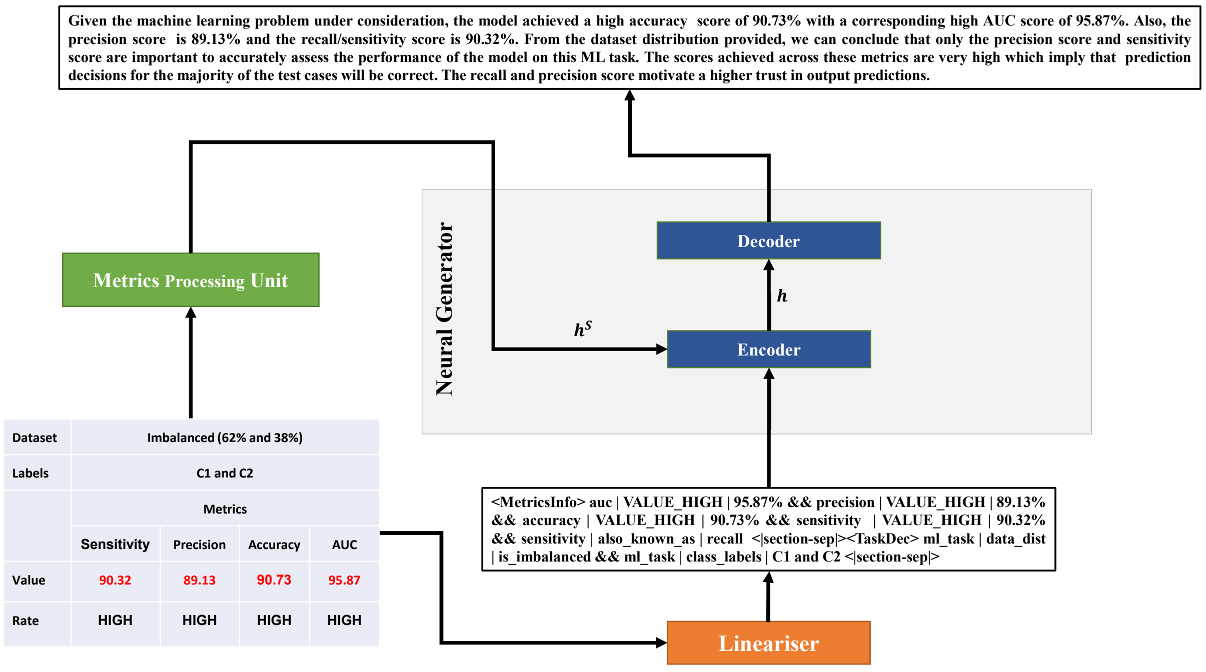
Textual Summary

Given the machine learning problem under consideration, the model achieved a high accuracy score of 90.73% with a corresponding high AUC score of 95.87%. Also, the precision score is 89.13% and the recall/sensitivity score is 90.32%. From the dataset distribution provided, we can conclude that only the precision score and sensitivity score are important to accurately assess the performance of the model on this ML task. The scores achieved across these metrics are very high which imply that prediction decisions for the majority of the test cases will be correct. The recall and precision score motivate a higher trust in output predictions.

Figure 2: An example of a classification performance report table (containing the score with respect to each metric along with information on the distribution of the underlying data across the two classes: C1 and C2) and the corresponding textual explanation.

b Discuss the overall performance of the model as shown by the values of the evaluation metrics. (Your answer should capture the implications of achieving such scores across the different metrics.)

When tasked to summarise information in numerical tables, humans typically perform numerical reasoning where the values are rated as either high or low or moderate. These ratings are used as a guide to accurately present the implications of the values in numerical tables. Therefore, during the annotation, the experts were asked to rate the scores across each metric (taking into account the distribution of the dataset across the classes) on a 3-point scale (Low, Moderate, and High). These ratings are used to enrich the metrics' tables (for example, see Fig. 2), providing the NLG model with a form of numerical reasoning.
We collected 1010 annotations from experts; each submission was analysed manually by comparing it to the corresponding table. Out of 1010 submissions, 825 accurately captured the information presented in the table. We sampled 100 table-explanation pairs randomly as the test set and the remaining 725 table-explanation pairs are used for the training set. Ideally, we expect the NLG model to generate similar narratives for a numerical table, irrespective of the order of the metrics in the linearised input. However, our preliminary analysis suggested that neural models tend to be sensitive to the order of the metrics; that is, the models generated



Given the machine learning problem under consideration, the model achieved a high accuracy score of 90.73% with a corresponding high AUC score of 95.87%. Also, the precision score is 89.13% and the recall/sensitivity score is 90.32%. From the dataset distribution provided, we can conclude that only the precision score and sensitivity score are important to accurately assess the performance of the model on this ML task. The scores achieved across these metrics are very high which imply that prediction decisions for the majority of the test cases will be correct. The recall and precision score motivate a higher trust in output predictions.

Dataset	Imbalanced (62% and 38%)			
Labels	C1 and C2			
	Metrics			
	Sensitivity	Precision	Accuracy	AUC
Value	90.32	89.13	90.73	95.87
Rate	HIGH	HIGH	HIGH	HIGH

```
<MetricsInfo> auc | VALUE_HIGH | 95.87% && precision | VALUE_HIGH | 89.13% && accuracy | VALUE_HIGH | 90.73% && sensitivity | VALUE_HIGH | 90.32% && sensitivity | also_known_as | recall <|section-sep|><TaskDec> ml_task | data_dist | is_imbalanced && ml_task | class_labels | C1 and C2 <|section-sep|>
```

Figure 3: An illustration of our table-to-text neural generator trained to produced the textual explanation based on the linearised table and metrics-ratings-values semantic representation, h^s, learned by the Metrics Processing Unit (MPU).

different narrations for the same table depending on the order of the metrics. Even though these narrations were fluent, they sometimes contained incorrect facts when compared to the related tables. Therefore, to make the models less dependent on the order of the metrics and focus more on verbalising the facts presented in the table, the training set is augmented with new table-explanation pairs from the permutations of the order of the metrics. This process increased the training set size from 725 to 4529 table-explanation pairs. Table 1 provides some statistics about the dataset. In Section 6, we compare the performance of the neural generators trained on the original dataset to those trained on the permuted dataset.

4. Models

4.1. Problem Definition

In this work, the input to the NLG models is a numerical table containing the evaluation metrics' scores of a classifier, the list of class labels and the distribution of the dataset across the labels. Specifically, the metrics table summarising the classifier's performance on a given machine learning problem is represented as $T = [D, C, S]$, where $D \in \{is\_balanced, is\_imbalanced\}$ is a flag indicating if the dataset was balanced, $C = [C_1, C_2, \cdots, C_k]$ is the list of class labels, and $S = [(m_1, r_1, v_1), (m_2, r_2, v_2), \cdots, (m_n, r_n, v_n)]$ is the list of performance scores. The performance scores S consist of the metrics $[m_1, m_2, \cdots, m_n]$ their values $[v_1, v_2, \cdots, v_n]$ and the annotator's ratings $[r_1, r_2, \cdots, r_n]$ where n is the number of metrics. The goal is to generate an analytical textual explanation $Y = (y_1, y_2, \cdots, y_b)$ of length b based on the information presented in T, where y_t is the t^{th} target word. It is noteworthy that the generated text Y should be fluent and numerically supported by T. An example of a table and the analytical textual explanation is shown in Fig. 2.

The neural generation model learns its parameter θ by maximising the likelihood function:

$$\mathsf{P}(Y \mid T; \theta) = \prod_{t=1}^{b} \mathsf{P}(y_t \mid y_{<t}, T; \theta)$$

where $y_{<t} = y_1, \cdots, y_{t-1}$ is the partial target sequence generated.

4.2. Baselines

Given the limited amount of data, experiments are conducted using pre-trained language models, T5 (Raffel et al., 2020) and BART (Lewis et al., 2020). Leveraging these pre-trained language models has been shown to produce high quality texts even when fine-tuned on limited amount of data (Peng et al., 2020; Su et al., 2021; Suadaa et al., 2021).

T5 model T5 (Raffel et al., 2020) is a transformer-based model trained in a multitask fashion on a variety of unsupervised and supervised NLP tasks including summarisation, classification, and translation. This neural model treats all text-based language problems as a text-to-text generation task (Raffel et al., 2020).

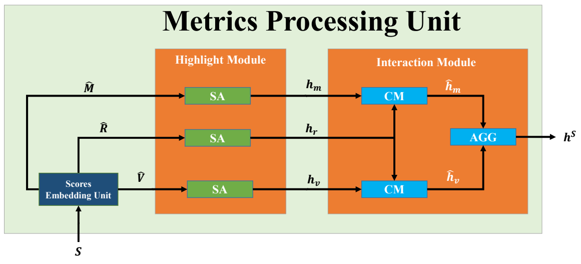


Figure 4: Architecture of the Metrics Processing Unit (MPU) employed to learn to h^s from the list of performance scores S.

BART model BART (Lewis et al., 2020) is a denoising auto-encoder for pre-training sequence to sequence models. This transformer-based architecture was trained to reconstruct the original text from corrupted input text. BART exploits the strengths of BERT (Devlin et al., 2019) and GPT (Radford et al., 2019). Specifically, it comprises a bidirectional encoder (similar to BERT) and a left-to-right decoder (similar to GPT).

4.3. Table Representation

BART and T5 were not trained on any table-to-text generation task and as such, all text-based language problems using these models need to be framed as text-to-text generation problems (Chen et al., 2020b; Suadaa et al., 2021; Hoyle et al., 2021). In this study, the input to the neural NLG models is a table summarising the classification performance of a given classier on an arbitrary ML task, as shown in Fig. 2.

Therefore, in order to convert it to a text-to-text problem and following (Moryossef et al., 2019; Chen et al., 2020b; Suadaa et al., 2021), the input table is linearised into a flat string. In this paper, the linearisation is performed by concatenating metric information, class labels, and the dataset distribution based on the following template:
<METRICSINFO> m_1 | VALUE_r_1 | v_1 && m_2 | VALUE_r_2 | v_2 && $\cdots$ && m_n | VALUE_r_n | v_n <|SECTION-SEP|> <TASKDEC> ML_TASK | DATASET_ATTRIBUTES | D && ML_TASK | CLASS_LABELS | $C_1, C_2, \cdots$, AND C_k <|SECTION-SEP|> <|TABLE2TEXT|>

Fig. 3 shows an example of the input data and the corresponding linearised representation. The linearised data is tokenised as $X = [x_1, x_2, \cdots, x_p]$ and converted into the tokens embedding $E^x = \left[e^x_1, e^x_2, \cdots, e^x_p\right]$ by the encoder's embedding unit. p is the number of tokens in the linearised data. The encoder outputs the contextual representation of the linearised table, $h \in \mathbb{R}^{p \times d_{model}}$, based on E^x. The decoder iteratively attends to previously generated tokens $y_{<t}$ (via self-attention) and the encoder outputs h (via cross-attention) to predict the next target token, y_t.

4.4. Metrics Processing Unit

One drawback of using only the linearised data input is that it fails to capture the actual information and relations represented by the structured data (Suadaa et al., 2021; Mager et al., 2020; Hoyle et al., 2021). On this task, we would like the models' decoders to be able to accurately verbalise the metrics information. The performance of the decoder is linked to that of the encoder; hence, exploiting strategies to improve the data representation ability of the encoder can further improve the quality of the output summaries. To this end, we propose augmenting the linearised input with semantic representations of the metrics information generated by a neural module: *Metrics Processing Unit* (MPU). Fig. 3 illustrates how the MPU is integrated with a given pre-trained neural generator. The MPU as shown in Fig. 4 comprises three main parts: *Scores Embedding Unit*, *Highlight Module* and *Interaction Module*.

Scores Embedding Unit Given $S = [(m_1, r_1, v_1), (m_2, r_2, v_2), \cdots, (m_n, r_n, v_n)]$, this unit generates the subword token-based embeddings $\hat{M} \in \mathbb{R}^{n*z \times d_{model}}$, $\hat{R} \in \mathbb{R}^{n*z \times d_{model}}$ and $\hat{V} \in \mathbb{R}^{n*z \times d_{model}}$, respectively, representing the metrics, rate and values. z is the maximum number of subword tokens and d_{model} is the dimension of hidden representation. To reduce the size of the final model, the embeddings are performed using the same embedding parameters employed by the *Neural Generator*'s encoder and decoder subnetworks.

Highlight Module This module produces the semantic representations of the metric names, values and ratings from their corresponding subword tokens representation produced by the *Scores Embedding Unit*. Specifically, the inputs to this module are $\hat{M} \in \mathbb{R}^{n*z \times d_{model}}$, $\hat{R} \in \mathbb{R}^{n*z \times d_{model}}$ and $\hat{V} \in \mathbb{R}^{n*z \times d_{model}}$, the outputs of the *Scores Embedding Unit*. Three self-attention units (one for each input representation) are employed to transform $\hat{M}, \hat{V}$ and $\hat{R}$, respectively into $h_m \in \mathbb{R}^{n*z \times d_{model}}$, $h_v \in \mathbb{R}^{n*z \times d_{model}}$, and $h_r \in \mathbb{R}^{n*z \times d_{model}}$.

Interaction Module This module generates the representation $h^s \in \mathbb{R}^{n*z \times d_{model}}$, the metrics-values-ratings contextual information, from combination of outputs of the *Highlight Module*. As shown in Fig. 4, this module consists of two Combination (CM) units and one Aggregation unit (AGG). The output is computed as follows:

$$\begin{aligned} h^s &= \mathrm{AGG}(\hat{h}_m, \hat{h}_v) + h_m + h_v + h_r \\ \mathrm{AGG}(\hat{h}_m, \hat{h}_v) &= W_a \left[\hat{h}_m; \hat{h}_v\right] \\ \hat{h}_m &= \mathrm{CM}(h_m, h_r), \hat{h}_v = \mathrm{CM}(h_v, h_r) \\ \hat{h} &= \mathrm{CM}\left(A, B\right) \\ \mathrm{CM}\left(A, B\right) &= \mathrm{ReLU}\left(W_c\left[A; B\right]\right) + B \end{aligned}$$

where $W^a, W^c \in \mathbb{R}^{2 \cdot d_{model} \times d_{model}}$ are trainable model parameters employed to compute h^s. $[\cdot;\cdot]$ denotes concatenation operation. The generated $h^s = [h^s_1, h^s_2, \cdots, h^s_{n*z}]$ is concatenated with the embedding of the linearised input table (E^x), then processed via the different encoder layers to generate the contextualised joint source representations $h \in \mathbb{R}^{(n*z+p)\times d_{model}}$.

5. Experiments

5.1. Model Settings, Training and Inference

Different variants of the T5 and BART models are explored in this work: T5-small, T5-base, T5-large, BART-base, and BART-large. These models differ in terms of the number of parameters (model size). The fine-tuning of the models is performed using Adam optimizer (Kingma and Ba, 2014) with an initial learning rate equal to $3e^{-4}$. For the T5 models, the learning rate is scheduled linearly during the course of training without any warmup schedule steps. However, our preliminary experiments conducted on BART models showed that warmup steps are required to achieve good generation performance. Therefore, 21% of the training steps are used as the linear warmup schedule steps. For simplicity, the T5-variant+MPU and BART-variant+MPU respectively, denote the T5 and BART model variants augmented with the auxiliary information from the MPU. During inference, the textual explanations are generated via beam search with a beam size of 8, length penalty $\alpha = 8.6$, and repetition penalty of 1.5. Our models implementations are based on Huggingface Transformers (Wolf et al., 2019).

5.2. Evaluation Metrics

The generation performance of the models is evaluated based on the quality of the generated textual explanations. The metrics employed to assess the quality are BLEU (Papineni et al., 2002) , METEOR (Banerjee and Lavie, 2005), BLEURT (Raffel et al., 2020) and PARENT (Dhingra et al., 2019). BLEU[2], and METEOR compute the surface-level similarity between the generated texts and only the human (reference) texts. Dhingra et al. (2019) argued that BLEU, and METEOR sometimes penalise generated text for including additional information, correct according to the table but missing in the reference text. In response, they proposed PARENT, a data-to-text evaluation metric that takes into consideration the information present in the table. The BLEURT score is a semantic equivalence-based metric indicating the extent to which the generated text is fluent and conveys the meaning of reference text. For each metric mentioned above, the higher the score, the better the model.

Each model is trained five times with different random seeds and we report the generation performance of the models based on the average of the scores for each evaluation metric.

[2]computed with https://github.com/awslabs/sockeye/tree/master/contrib/sacrebleu

Model	BLEU	METEOR	PARENT	BLEURT
T5-small	37.83±2.25	39.46±1.83	32.12±1.16	55.52±0.22
T5-small + MPU	40.58±1.95	43.40±1.05	**34.04±0.65**	**56.15±0.40**
T5-base	46.31±0.89	47.45±0.35	30.99±0.7	54.09±0.30
T5-base + MPU	45.46±0.60	47.75±0.40	31.33±0.67	54.46±0.19
T5-large	45.70±0.73	47.11±0.31	31.08±0.63	54.53±0.2
T5-large + MPU	46.03±0.75	47.53±0.41	31.58±0.73	54.63±0.42
BART-base	44.83±1.03	**47.83±0.22**	32.79±0.86	54.71±0.16
BART-base + MPU	45.75±0.84	**47.83±0.64**	33.26±0.96	55.26±0.31
BART-large	45.57±1.96	46.41±1.44	32.29±2.12	51.29±1.76
BART-large + MPU	**46.98±0.73**	47.52±1.50	33.27±0.6	51.25±1.67

Table 2: Evaluation of generation performance of the neural models on the permuted dataset. "+ MPU" detonates training the variant of the pre-trained baselines with *Metrics Processing Unit* (MPU). The models are fine-tuned with five different random seed and the scores are based on the average and standard deviation.

Model	BLEU	METEOR	PARENT	BLEURT
T5-small	26.0±0.81	32.64±0.66	30.70±0.51	53.12±0.34
T5-small + MPU	26.70±1.12	33.34 ±1.14	29.39±1.24	53.69±0.72
T5-base	32.83±1.70	35.81±1.36	31.70±0.80	55.51±0.47
T5-base + MPU	32.59±3.6	36.80±1.53	30.36±1.8	55.14±0.36
T5-large	**35.11±2.93**	**38.38±1.69**	32.73±2.01	55.67±0.51
T5-large + MPU	32.27±2.40	36.69±1.11	32.58±2.61	55.84±1.04
BART-base	26.91±3.76	34.90±1.34	31.46±2.33	56.30±0.42
BART-base + MPU	28.79±2.84	35.59±0.93	29.71±3.10	55.13±0.61
BART-large	28.03±2.64	35.55±2.86	32.90±2.39	54.40±1.54
BART-large + MPU	30.13±1.8	38.15±1.93	**34.68±1.1**	**56.48±0.23**

Table 3: Evaluation of generation performance of the neural models on the original dataset (725 data-text training pairs). "+ MPU" detonates training the variant of the pre-trained baselines with *Metrics Processing Unit* (MPU).

6. Results

This section presents the evaluation performance of the models on the NLG task under consideration. Table 2 shows the scores achieved by the models across the evaluation metrics: BLEU, METEOR, PARENT, and BLEURT. The different variants of the pre-trained models achieved varying scores. Among the T5 models, the T5-small achieved the worst performance, according to BLEU, and METEOR. However, it outperformed T5-base and T5-large in terms of PARENT and BLEURT scores. For large T5 models (T5-base and T5-large), there is a higher possibility that the generated texts will be identical to the reference texts; hence, the high scores for the word-overlap metrics: BLEU, and METEOR. The PARENT score suggests the T5-small is very good at matching both the reference text and the source table. Between BART-base and BART-large, the former outperforms the later judging based on the METEOR, PARENT and BLEURT scores. On the other hand, BART-large scored +0.74 BLEU over BART-base.

Effect of Metric Processing Unit According to Table 2, augmenting the linearised representation with the semantic representations of the metrics information in the table further improves the generation performance of the underlying models. The T5-small ben-

Model	Generated
Table	<MetricsInfo>auc | VALUE_HIGH | 95.87% && precision | VALUE_HIGH | 89.13% && accuracy | VALUE_HIGH | 90.73% && sensitivity | VALUE_HIGH | 90.32% && sensitivity | also_known_as | recall <|section-sep|><TaskDec>ml_task | data_dist | is_imbalanced && ml_task | class_labels | C1 and C2 <|section-sep|> <|table2text|>
Reference	Given the machine learning problem under consideration, the model achieved a high accuracy score of 90.73% with a corresponding high AUC score of 95.87%. Also, the precision score is 89.13% and the recall/sensitivity score is 90.32%. From the dataset distribution provided, we can conclude that only the precision score and sensitivity score are important to accurately assess the performance of the model on this ML task. The scores achieved across these metrics are very high which imply that prediction decisions for the majority of the test cases will be correct. The recall and precision score motivate a higher trust in output predictions.
T5-small	The performance of the classifier on this binary classification task as evaluated based on the precision, AUC, accuracy, and sensitivity achieved the scores 89.13%, 90.32% , 95.87%, and 90.73% respectively. These scores support the conclusion that this model will be highly effective at correctly classifying most test cases/samples with only a few misclassifications. Furthermore, the confidence in predictions related to the label C2 is very high.
T5-small+MPU	The algorithm's ability to correctly classify test samples as either C1 or C2 was assessed based on the metrics accuracy, AUC, precision, and sensitivity. It scored 90.73% , 95.87%, 89.13% and 90.32%, respectively. As shown in the table, we can confirm that the algorithm has a very high classification performance and will be very effective at correctly labeling most test cases drawn from any of the different classes.
T5-base	On this imbalanced classification task, the trained model scored 90.32% (recall), 90.73% (accuracy), and 95.87% (AUC). From the precision score, it is obvious that the model has a moderately high false-positive rate hence the confidence in predictions related to the C2 label is very high. However, based on the sensitivity and precision scores, we can see that some examples belonging to C1 are likely to be misclassified as C2. This is further supported by the high accuracy and AUC scores.
T5-base+MPU	The classifier scored close to perfect scores across all the metrics (i.e. Precision, AUC, Accuracy and Sensitivity). From the results table, we can see that it scored 90.73% (accuracy), 90.32% (sensitivity), and 95.87% (AUC). Surprisingly, these scores were achieved even though the dataset was imbalanced. With such high scores for precision and sensitivity, the model is shown to have a lower misclassification error rate. Overall, this model has relatively high classification performance, and hence will struggle to correctly identify the labels for only a few test cases belonging to the different classes.
T5-large	Evaluating the performance of the model on this classification task produced the scores: 95.87% for AUC, 90.73% for accuracy, 89.13% precision, and 90.32% for sensitivity (recall). From the recall and precision scores, we can see that the classification algorithm is very effective at correctly classifying most unseen test cases or samples with only a small margin of error (the misclassification error rate is 9.27%). The very high accuracy coupled with the very low precision score demonstrates its capability to correctly identify the correct class labels for several test instances.
T5-large+MPU	The classifier scored close to perfect scores across all the metrics (i.e. Precision, AUC, Accuracy and Sensitivity). From the results table, we can see that it scored 89.13% (Precision), 90.32% (sensitivity), 95.87% (AUC), and 90.73%(Accuracy). Since the dataset was imbalanced, it would be wise to analyze prediction performance based on the balance between the recall and precision scores. The precision and sensitivity scores show how good the model is at correctly recognising the observations under the different classes (C1 and C2). In summary, the models are likely to have a lower misclassification error.
BART-base	The classification performance scores achieved by the model on this binary classification task are as follows: (1) AUC score of 95.87, (2) Accuracy equal to 90.73%, (3) Precision score equal 89.13%, and (4) Sensitivity (sometimes referred to as the recall) score is 90.32%. These scores across the different metrics suggest that this model is very effective and can accurately identify the true labels for several test cases/samples with a small margin of error (actually, the misclassification error is 9.27%)
BART-base+MPU	The scores achieved by the AI algorithm on this binary classification task are as follows (1) AUC score of 95.87%, (2) Accuracy equal to 90.73%, and (3) Precision score equal 89.13%. These scores across the different metrics suggest that this model is very effective and can accurately identify the true labels for the majority of the cases with a small margin of error. Furthermore, the precision and recall scores indicate that the likelihood of misclassifying C1 cases as C2 is very marginal (that is, it has a very low false-positive rate).
BART-large	The performance of the model on this binary classification task as evaluated based on the precision, AUC, accuracy, and sensitivity scored 89.13%, 95.87%, 90.73%, and 90.32% respectively, implying that it is a very effective model. These scores indicate that the likelihood of misclassifying test samples is very marginal. However, the scores were expected the dataset was perfectly balanced between the two class labels C1 and C2.
BART-large+MPU	On this imbalanced classification task, the trained model reached an AUC score of 95.87, an accuracy of 90.73, with a precision and sensitivity scores equal to 89.13 and 90.32, respectively. These results/scores are very impressive as one can conclude that this model is almost perfect with higher confidence in its prediction decisions. In summary, only a small number of test cases are likely to be misclassified as indicated by the accuracy, sensitivity, and precision.

Table 4: Example of the model performance narrations generated by the fine-tuned T5 and BART models based on the table data (converted into flat-string). Sentences and phrases conveying correct information according to the related table are highlighted in green, while incorrect ones are marked in red.

efited most among all the models. Specifically, there is +2.75, +3.94, +1.92, and +0.6 increase in the BLEU, METEOR, PARENT and BLEURT, respectively. Furthermore, T5-small+MPU is shown to have the best match to both the reference text and the source table according to PARENT. It outperforms the next best models BART-base+MPU and BART-large+MPU by +0.77 PARENT, and +0.78 PARENT, respectively. Furthermore, despite having a poor surface-level match to the reference texts according to BLEU and METEOR, it achieves the best BLEURT score meaning it's outputs are fluent and semantically equivalent to the reference texts.

Permuted vs Original To analyse the impact of augmenting the dataset with the diverse representations of the tables by permuting the order of the metrics, we trained all the neural models on the original 725 training pairs. The generation performance of the neural models is summarised in Table 3. Compared to the results in Table 2, the models performed worse in general according to the BLEU and METEOR scores. For example, while BART-large scored 28.09 BLEU, and 35.55 METEOR when trained on the original dataset, it scored 45.57 BLEU and 46.41 METEOR when trained on the permuted dataset. Conversely, in some cases, the model trained on the original dataset outperformed the corresponding variant trained on the permuted dataset in terms of the PARENT and BLEURT scores. Judging based on the average performance across all the metrics, the augmentation of the dataset is shown to improve generalization performance given that the models will be exposed to diverse representations of the tables.

Quality Analysis Table 4 shows the performance textual explanations generated by the models under consideration based on the table shown in Fig. 2. Sentences and phrases conveying correct information according to the related table are highlighted in green, while incorrect ones are marked in red. As shown, the generators are able to produce high-quality classification performance summaries capturing the information presented in the input structured data. However, there were a number of cases where the generators failed to accurately verbalise the content of the related performance metric table. The errors are mainly from the models trained without MPU and among these models, only BART-base produced a correct verbalisation of the input table. The summary from T5-small is mostly valid; however, the metrics (AUC, accuracy and sensitivity) and their corresponding scores are mentioned in the wrong order. The T5-base made an incorrect assessment of the "precision" and "recall" scores when it concluded that the "false-positive rate" is moderately high. In the case of the T5-large, it stated that the precision is very low even though it was rated "HIGH". BART-large produced a wrong statement about the distribution of the dataset between the classes, C1 and C2. Augmenting the linearised representations with the metrics-values-ratings, contextual information from MPU allow the T5 and BART models to generate accurate analytical textual explanations based on the related table.

7. Conclusion

This work presents a new NLG dataset for generating textual explanations describing the performance of classification models. Presenting the generated texts along with the numerical tables will allow for a better understanding of the classification performance of ML models. We trained baselines by fine-tuning state-of-the-art pre-trained models: T5 and BART. Experimental results show the feasibility of utilising these large pre-trained language models to generate fluent and accurate statements based on structured data. However, analyses suggest that neural models in some instances produce statements containing wrong information according to the input table. This weakness can be attributed to direct linearisation of the input tables. To address this problem, we introduced the *Metric Processing Unit* (MPU) which, when combined with the linearised input, produced the best performance across the different T5 and BART variants.

In the future, we plan on conducting in-depth human evaluations to assess the quality and usefulness of the generated texts. As stated in Section 3 only the most popular evaluation metrics were considered hence the NLG models will struggle to produce meaningful analytical texts for instances with less common metrics such as Cohen Kappa, Matthews Correlation Coefficient (MCC), and Brier score. Therefore a possible avenue of future is expanding the dataset to include textual explanations on these metrics.

8. Bibliographical References

Banerjee, S. and Lavie, A. (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.

Banik, E., Gardent, C., and Kow, E. (2013). The kbgen challenge. In *Proceedings of the 14th European Workshop on Natural Language Generation*, pages 94–97.

Chemmengath, S., Kumar, V., Bharadwaj, S., Sen, J., Canim, M., Chakrabarti, S., Gliozzo, A., and Sankaranarayanan, K. (2021). Topic transferable table question answering. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 4159–4172.

Chen, D. L. and Mooney, R. J. (2008). Learning to sportscast: a test of grounded language acquisition. In *Proceedings of the 25th international conference on Machine learning*, pages 128–135.

Chen, W., Chang, M.-W., Schlinger, E., Wang, W. Y., and Cohen, W. W. (2020a). Open question answering over tables and text. In *International Conference on Learning Representations*.

Chen, W., Chen, J., Su, Y., Chen, Z., and Wang, W. Y. (2020b). Logical natural language generation from open-domain tables. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7929–7942.

Chen, Z., Eavani, H., Chen, W., Liu, Y., and Wang, W. Y. (2020c). Few-shot nlg with pre-trained language model. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 183–190.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2019). Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT (1)*.

Dhingra, B., Faruqui, M., Parikh, A., Chang, M.-W., Das, D., and Cohen, W. (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.

Goldberg, E., Driedger, N., and Kittredge, R. I. (1994). Using natural-language processing to produce weather forecasts. *IEEE Expert*, 9(2):45–53.

Hoyle, A. M., Marasović, A., and Smith, N. A. (2021). Promoting graph awareness in linearized graph-to-text generation. *ArXiv*, abs/2012.15793.

Kingma, D. P. and Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv:1412.6980*.

Koncel-Kedziorski, R., Bekal, D., Luan, Y., Lapata, M., and Hajishirzi, H. (2019). Text generation from knowledge graphs with graph transformers. 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 2284–2293.

Kukich, K. (1983). Design of a knowledge-based report generator. In *21st Annual Meeting of the Association for Computational Linguistics*, pages 145–150.

Lebret, R., Grangier, D., and Auli, M. (2016). Neural text generation from structured data with application to the biography domain. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1203–1213.

Lewis, M., Liu, Y., Goyal, N., Ghazvininejad, M., Mohamed, A., Levy, O., Stoyanov, V., and Zettlemoyer, L. (2020). Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880.

Liang, P., Jordan, M. I., and Klein, D. (2009). Learning semantic correspondences with less supervision. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 91–99.

Liu, T., Wang, K., Sha, L., Chang, B., and Sui, Z. (2018). Table-to-text generation by structure-aware seq2seq learning. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Mager, M., Fernandez Astudillo, R., Naseem, T., Sultan, M. A., Lee, Y.-S., Florian, R., and Roukos, S. (2020). GPT-too: A language-model-first approach for AMR-to-text generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1846–1852, Online, July. Association for Computational Linguistics.

McKeown, K. (1992). *Text generation*. Cambridge University Press.

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

Novikova, J., Lemon, O., and Rieser, V. (2016). Crowd-sourcing nlg data: Pictures elicit better data. In *Proceedings of the 9th International Natural Language Generation conference*, pages 265–273.

Novikova, J., Dušek, O., and Rieser, V. (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.

Papineni, K., Roukos, S., Ward, T., and Zhu, W.-J. (2002). Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*, pages 311–318. ACL.

Parikh, A., Wang, X., Gehrmann, S., Faruqui, M., Dhingra, B., Yang, D., and Das, D. (2020). Totto: A controlled table-to-text generation dataset. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1173–1186.

Peng, B., Zhu, C., Li, C., Li, X., Li, J., Zeng, M., and Gao, J. (2020). Few-shot natural language generation for task-oriented dialog. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings*, pages 172–182.

Puduppully, R., Dong, L., and Lapata, M. (2019). Data-to-text generation with content selection and planning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 6908–6915.

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., Sutskever, I., et al. (2019). Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., and Liu, P. J. (2020). Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21:1–67.

Reiter, E. and Dale, R. (1997). Building applied natural language generation systems. *Natural Language Engineering*, 3(1):57–87.

Ribeiro, L. F., Schmitt, M., Schütze, H., and Gurevych, I. (2020). Investigating pretrained language models for graph-to-text generation. *arXiv preprint arXiv:2007.08426*.

Strauss, M. and Kipp, M. (2008). Eric: a generic rule-based framework for an affective embodied commentary agent. In *Proceedings of the 7th international joint conference on Autonomous agents and multiagent systems-Volume 1*, pages 97–104.

Su, Y., Meng, Z., Baker, S., and Collier, N. (2021). Few-shot table-to-text generation with prototype memory. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 910–917.

Suadaa, L. H., Kamigaito, H., Funakoshi, K., Okumura, M., and Takamura, H. (2021). Towards table-to-text generation with numerical reasoning. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1451–1465.

van der Lee, C., Krahmer, E., and Wubben, S. (2017). Pass: A dutch data-to-text system for soccer, targeted towards specific audiences. In *Proceedings of the 10th International Conference on Natural Language Generation*, pages 95–104.

Wang, H., Zhang, X., Ma, S., Sun, X., Wang, H., and Wang, M. (2018). A neural question answering model based on semi-structured tables. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1941–1951.

Wen, T.-H., Gasic, M., Mrksic, N., Su, P.-h., Vandyke, D., and Young, S. J. (2015). Semantically conditioned lstm-based natural language generation for spoken dialogue systems. In *EMNLP*.

Wiseman, S., Shieber, S. M., and Rush, A. M. (2017). Challenges in data-to-document generation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2253–2263.

Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., Cistac, P., Rault, T., Louf, R., Funtowicz, M., et al. (2019). Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*.