Text Generation for Populating Semi-structured Tables

by
Priya Mehta

Under the Supervision of:
Dr. Vikram Goyal and Dr. Tanmoy Chakraborty

Submitted
in partial fulfillment of the requirements for the degree of
Master of Technology

to

Indraprastha Institute of Information Technology Delhi
December, 2021

Certificate

This is to certify that the thesis titled **"Text Generation for Populating Semi-structured Tables"** being submitted by **Priya Mehta** to the Indraprastha Institute of Information Technology Delhi, for the award of the Master of Technology, is an original research work carried out by him under my supervision. In my opinion, the thesis has reached the standards fulfilling the requirements of the regulations relating to the degree.

The results contained in this thesis have not been submitted in part or full to any other university or institute for the award of any degree/diploma.

May, 2022   Dr Vikram Goyal   Dr. Tanmoy Chakraborty

Department of Electronics and Communication Engineering
Indraprastha Institute of Information Technology Delhi
New Delhi 110 020

Acknowledgements

First and foremost, I would like to thank my advisors for the continuous support and belief in our work. I would also like to thank my colleagues Meet Maheshwari, Yash Kumar Atri and Sarah Masud for their support and suggestions throughout the whole period.

Abstract

Heterogeneous semi-structured tables are commonly used to represent data on the internet. Recent years have seen a flurry of works in tasks that endeavor to comprehend such tabular information, such as table summarization, tabular question answering, and tabular fact-checking, to name a few. In this work, we proffer a new task in the realm of tabular data analysis called '*Populating Semi-structured Tables*', wherein, given a partially filled table and related content, the aim is to generate text for the missing cells in the table. While most of the tasks that reason over semi-structured tables utilize the transformer-based sequence-to-sequence models, the table's hierarchical structure and long-tailed nature seem to limit the performance of language models. Thus, we extend the traditional sequence-to-sequence models and propose sequence to multi-sequence models to handle multiple missing cell contents which are partially dependent on each other. Our inspiration comes from the system used for one-to-many sequence transduction problems with speech data which is yet to be experimented with for natural language generation tasks. The results show that our model, '**Mu**ltiple **Ce**ll **F**iller' ($MuCeF$) is better than the top baseline by a 15.44 ROUGE score and 34.54 METEOR score. Resources related to this work will be open-sourced for further research.

Contents

Certificate **i**

Acknowledgements **ii**

Abstract **iii**

List of Figures **v**

List of Tables **vi**

1 Introduction **1**
1.1 Background 1
1.2 Motivation 2
1.3 Use cases 3

2 Related Work **4**
2.1 Text Generation 4
2.2 Data-to-Text Generation 4

3 Dataset **6**
3.1 Original Dataset 6
3.2 Modified Dataset 7

4 Methodology **9**
4.1 Data Representation 9
4.2 Terminology 11
4.3 Problem Formulation 11
4.4 Model Architecture 12

5 Results **15**
5.1 Baseline and Model Comparison 15
5.2 Error Analysis 15

6 Conclusion **18**

Bibliography **19**

List of Figures

1.1 Input and Output dataset sample. 2

3.1 Topic distribution of the dataset. 7
3.2 Missing cell distribution in the train set. 8
3.3 Missing cell distribution in the test set. 8

4.1 Mapping Approaches 12
4.2 Model Architecture 13

5.1 Output Comparison between Ground Truth, MuCeF and TAPAS. . . . 17

List of Tables

3.1 Dataset Statistics. 7
3.2 Missing Cell Statistics. 8

4.1 Special tokens used in MuCeF. 11

5.1 Baselines and model comparison. 15
5.2 Analysis 16
5.3 Comparing ground truth with output from MuCeF and the best baseline 16

Chapter 1

Introduction

1.1 Background

Text generation is a long studied problem in Natural Language Processing [1] to seek solution to the problem of deciding what to communicate and how to organize it in the best possible manner. A sub-problem of Natural Language Generation (NLG) is Data-to-Text Generation which includes the task of generating a target free text description conditioned on source content in the form of structured data such as a table. Some examples include generating basketball game summaries given boxscore statistics [2] or generating line description of from tables especially including certain highlighted cells [3] or generating description from biographical data [4] and many others.

In Natural Language Understanding setting, language generation models are generally tested for their fluency and faithfulness to the task at hand. Moreover, structured data can also be used for testing model's ability to reason and infer over relationships between different records and attributes. Generation ability requires a mix of worldly knowledge, or common sense, and signals in form of context to prompt the model in the right direction. It is often seen that the model is able to produce grammatically correct sentences but are factually incorrect. Such problem is known as hallucination [5]. Hallucination is prominently seen when the training corpus contains noisy data residing due sub-optimal heuristics used while collecting the dataset.

In this work, we study a variation of data-to-text problem where the task is to generate target textual content for populating empty cells in the source partial table. By partial, it is meant that the table has few missing cell values and is not complete. The number of missing cell values can vary across data samples. For experimentation, our

Fig. 1.1. Input and Output dataset sample.

test field is tables scraped from Wikipedia. Web tables are semi-structured as they do not reside in proper relational database but have some organizational properties for representing the data. There are no strict restrictions on column type values or cell spanning. Since the web tables are scraped from Wikipedia, we have additional metadata corresponding to each table like the Wikipedia page title, section title, table caption etc which may provide some necessary context related to the information in the table. An example of input and output data is seen in figure 1.1.

1.2 Motivation

In general, most of the NLG tasks can be treated as an input sequence to an output sequence conversion problems. For example, machine translation has sequences from different languages, summarization has varied length of sequences, style transfer has different organization and vocabulary in input and output sequences etc. Well established neural sequence-to-sequence models [6] consisting of RNN/LSTM/GRU units are used to successfully handle these kind of problems. Here, for the problem of table filling, the partial table can be linearized and treated as one input sequence but the output contains a number sequences representing the content of missing cells. Also note that the number of output sequences is not fixed as the number of missing cells in the partial table can vary. Hence, for this work, simply treating the problem as a sequence to sequence learning does not suffice.

One may argue that, for the sake of simplicity, all the missing cell values can be merged into a single output sequence. But that thinking is flawed as different cell values can be missing from any position in the table and are not ordered. If they aren't ordered, there is no point in arranging them as a sequence. Even if we do so, the model may try to learn the ordering among these cell values which is unnecessary. Thus, there arises a need to study sequence-to-multi-sequence learning task. Our motivation comes from the sequence-to-multi-sequence learning [7] used in audio related tasks such as speech separation and multi-speaker speech recognition.

1.3 Use cases

Representing data in a tabular form is fairly common across all domains. Huge amount of knowledge is stored in tables across the web as it presents an organised manner to store and retrieve information quickly. Such web tables may often contain missing values. Another use case is to provide auto-complete suggestions to the user while filling up data in the table. While the commonly used tools support simple numerical value prediction like increment for serial number or formula based calculation specified by the user, they are unable to predict textual values. A system is needed which can not only take the partial table but also the accompanying textual information or metadata to generate most likely content to be filled in those missing cells. The complexity of web tables needs to be considered unlike relational tables like – 1. Cells can span multiple columns and rows, making the table structure inconsistent across different samples, 2. The values from the same column may not follow a single data type and can be heterogeneous, 3. Row headers are present along with column headers, 4. Tables are from a vastly diverse set of topics.

Note that MuCeF is not trained to predict future values that are yet to happen. It can only generate data values for which the context is given in form of input table/metadata or description.

Chapter 2

Related Work

2.1 Text Generation

Early works for text generation problems such as dialogue systems, machine translation, summarisation etc heavily relied on templates [8] and hand-engineered features [9] [10]. Slowly, template based models started getting replaced by trainable components [11]. Copy mechanism, a technique to selectively repeat phrases from input sequence as often done by humans in conversation, is also adopted in many works such as [12], [13], [14]. More recently, pre-trained language models (PLMs) have been largely adopted [15], [16] for various NLG benchmarks such as [17] for WebNLG challenge [18] as well as E2E challenge [19].

2.2 Data-to-Text Generation

Data to text generation tasks involves generating natural language descriptions for structured data like tables which is a long-studied problem. One of the early pipeline system [20] explicitly divides the task into content selection, planning and surface realization. A number of challenges have been proposed for data to text generation like sports game summary from table of game score statistics [21] and generating text from RDF triplets [22]. High performing models for data-to-text generation include language models specialized for handling tabular data such as [23] for text generation from table, [24] for table based question answering, [25] for table based fact verification etc. A related task introduced recently is conditional text generation based on highlighted table values in [3] tackled using PLMs in [26]. Other kind of approaches study how the generation problem can be tackled using pipeline system like traditional methods where content selection, planning and surface realization is done in order [27], [28], [29] [30].

Chapter 3

Dataset

3.1 Original Dataset

Originally ToTTo is proposed as a controlled table to text generation dataset. It a clean and annotator revised dataset with tables scraped from Wikipedia. The goal of the task is given the table with a few highlighted cells, table metadata,and set of highlighted cells, to produce the one-sentence description of the table conditioned on the highlighted cells. The purpose to provide highlighted table cells is to set a benchmark for high precision text generation. Using the highlighted content, the system should aim to generate text faithful to the source table to eliminate hallucination.

Main advantages of choosing ToTTo Dataset are as follows,

- Diverse topic distribution as seen in figure 3.1
- Complex table structure
- Monolingual (mainly in English) as our main aim is understanding tabular data and not multiple languages
- Noise free and open domain dataset

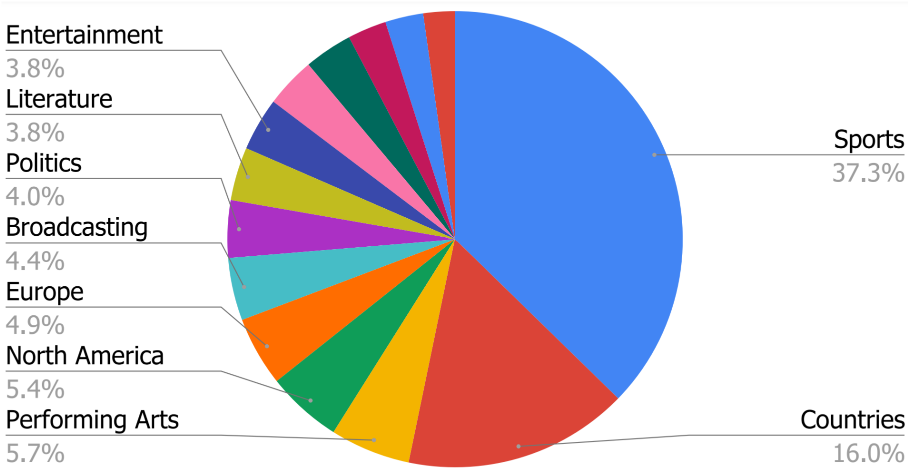

Fig. 3.1. Topic distribution of the dataset.

3.2 Modified Dataset

The task we wish to tackle is related to the task of controlled text generation. Our goal is to populate missing cell content on the table with the help description and metadata. We replace the highlighted cells of the original ToTTo dataset with special token <MCV> as described in 4.1. In 69.7% samples, missing cell content is supported by the description of table. Note that only regular cells are missing and not the headers. On average there are 3.55 missing cells while the maximum and minimum number of missing cells are 640 and 1 respectively. This shows the extreme variety of data samples in the dataset. The distribution of missing cells in training and testing dataset as shown in figure 3.2 and figure 3.3 is quite similar. These plots present distribution for tables with missing cells ranging from 1 to 10 and covers more than 98% of the whole dataset. Other useful statistics related to the dataset are given in table 3.1. Additionally, statistics specific to the required output is given in table 3.2.

Property	**Value**
Training set size	1,20,761
Test set size	7,700
Unique Tables	83,141
No. of Target tokens	941,950
Target Vocabulary Size	168,092

Table 3.1: Dataset Statistics.

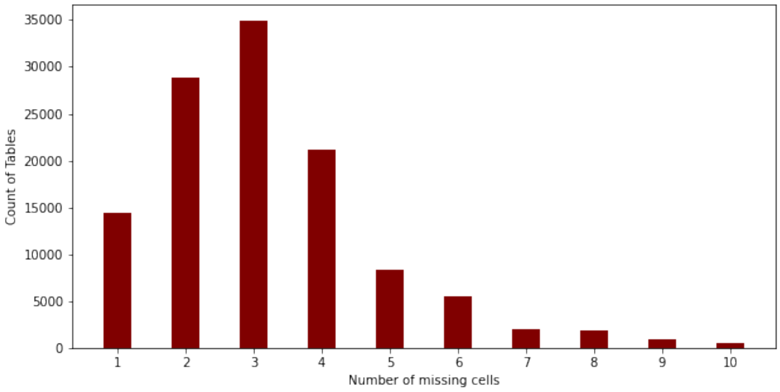

Fig. 3.2. Missing cell distribution in the train set.

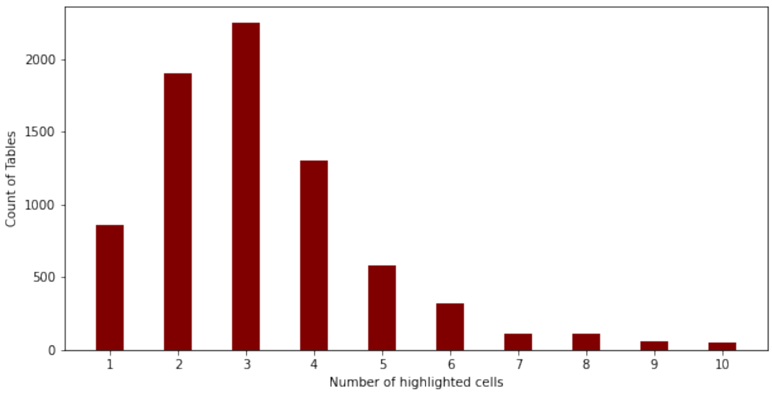

Fig. 3.3. Missing cell distribution in the test set.

Property	**Median**	**Average**	**Maximum**	**Minimum**
Cells per table	87	206.6	30,464	1
No. of Missing Cells	3	3.55	640	1
Missing Cell Length in characters	9	12.6	605	1
Empty cell percentage in table	3.42	7.457	100.0	0.0052

Table 3.2: Missing Cell Statistics.

Chapter 4

Methodology

4.1 Data Representation

1. Table -

To address the complexity of table structure, such as multiple/ none row headers and column headers (as shown in figures) as well as multiple row/column cell span, it is best to represent the table at cell level, instead of row level. When we linearize a table, structural information like position of the cell is lost so it is necessary to provide position separately for each cell. The input partial table is denoted as a sequence of cells in a top-to-down and left-to-right fashion as follows,

$$T = cell_1, cell_2, cell_3, \ldots., cell_x \tag{4.1}$$

where cell c_i of table T is further represented as

$$\begin{aligned} cell_i = (value, row_{header}, col_{header}, \\ row_{index}, col_{index}) \end{aligned} \tag{4.2}$$

where $value$ is the cell content or a special token <MCV>,

row_{header} is the row header of the row that cell belongs to, if any

col_{header} is the column header of the row that cell belongs to, if any

row_{index} is the row index of the row that cell belongs to

col_{index} is the column index of the row that cell belongs to

For the entire table, total number of cells $x = m * n$, where m and n are the number of rows and columns respectively. But for the sake of efficiency in time and performance, we consider only a subset of table that includes missing cell values. This makes it easier for the model to only focus on relevant content but limits the ability to perform reasoning in the context of the table structure. For subtable, the total number of cells $x =$ number of MCV tokens $< m * n$. It is also important to limit the length of input because in many transformer models like BERT, the input size is fixed to 512/1024 tokens, beyond which the model's performance degrades.

2. Metadata -

The second input is metadata corresponding to the table. The dataset contains many kinds of metadata information like page title, section title, and up to the first 2 sentences of the section text. For our task we limit it to two important labels, namely, page title indicating the wikipedia page the table belongs to and the section title indicating the section name inside the page that the table belongs to.

$$M = (m_{page-title}, m_{section-title}, m_{section-text}) \tag{4.3}$$

3. Text Description of Table -

The third input is text of one or two sentences S describing the table, especially taking the missing cell content into consideration.

4. Missing Cell Content -

The output text will contain x sequences denoting content of each missing cell,

$$O = o_1, <\text{MSS}>, o_2, <\text{MSS}>, o_3, \ldots, <\text{MSS}>, o_x \tag{4.4}$$

where each output sequence of variable length l as,

$$o_i =< \text{SOS} > w_1 w_2 w_3 \ldots w_l < \text{EOS} > \tag{4.5}$$

4.2 Terminology

Special tokens and their usage in the model is as described in the table. The last two tokens are introduced by us for this task.

Special Token	Usage
CLS	The classifier token which is used when doing sequence classification. It is the first token of the sequence.
SEP	The separator token, which is used when building a sequence from multiple input sequences. It is the last token of a sequence.
UNK	The unknown token is used to replace the rare words that do not fit in vocabulary.
PAD	Padding token is used to shape input in a batch to the same length.
SOS	Denotes start of the whole sequence.
EOS	Denotes end of the whole sequence.
MCV	The missing cell value token inside the vector indicates the position of the missing cell.
MSS	The multi-sequence separator token is used to separate various output sequences.

Table 4.1: Special tokens used in MuCeF.

Note that here the <MCV> token differs from the traditional <MASK> token used in Language Modelling because the mask token corresponds to exactly one token from the input whereas the missing cell value token corresponds to an unknown number of input tokens. Another thing to note is that <MSS> special token is required to separate multiple sequences as the output may contain new line.

4.3 Problem Formulation

Table Filling task takes a partially complete table along with corresponding metadata and text description as input. The target is to produce content for all the missing cells as shown in the figure. Thus, the problem is defined as a multi-input problem consisting of semi-structured as well as free text data to produce multiple meaningful text sequences to fit the table well. We define the problem as each data sample

d containing input triplet – partial table T, metadata M and description S, to be mapped to variable number of output sequences O as,

$$d(T, M, S) \longrightarrow O \tag{4.6}$$

We performed few experiments by excluding metadata as it is not an integral part of input data but the results were lower compared to subtable plus metadata as input. So in the further experiments all three input modalities will be considered.

4.4 Model Architecture

Sequence mapping approaches [7] are broadly of three kinds as described in figure 4.1, wherein, the input sequence remains fixed, but the relation among output sequences differ. As depicted in 4.1. A, a parallel mapping approach can be seen in text auto-completion based upon input prompt, in which generated suggestions are entirely independent of each other. An example of figure 4.1. B, serial mapping is to convert audio language using speech recognition and then machine translation to get the final output sequence. As imaginable, an intermediate output sequence is fully utilized to obtain the final output sequence. Final approach figure 4.1. C is the conditional mapping where the subsequent output sequences partially depend on the prior output sequences.

The last approach seems fitting for the problem defined in this work of multiple cell filling in the table as the cells belonging to same table (maybe same record or attribute) may have a partial relation, rather than being completely dependent on each other. Our effort lies in capturing this relationship information which may provide better context along with the required inputs described in section 4.1.

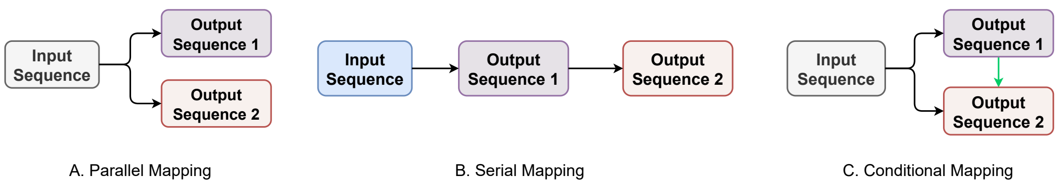

Fig. 4.1. Mapping Approaches

Warm-starting the encoder-decoder model is defined as composing an encoder-decoder model of pre-trained stand alone model checkpoints [31]. Although a number of models such as BERT, GPT, RoBERTa can be used; for our experiments we initialize

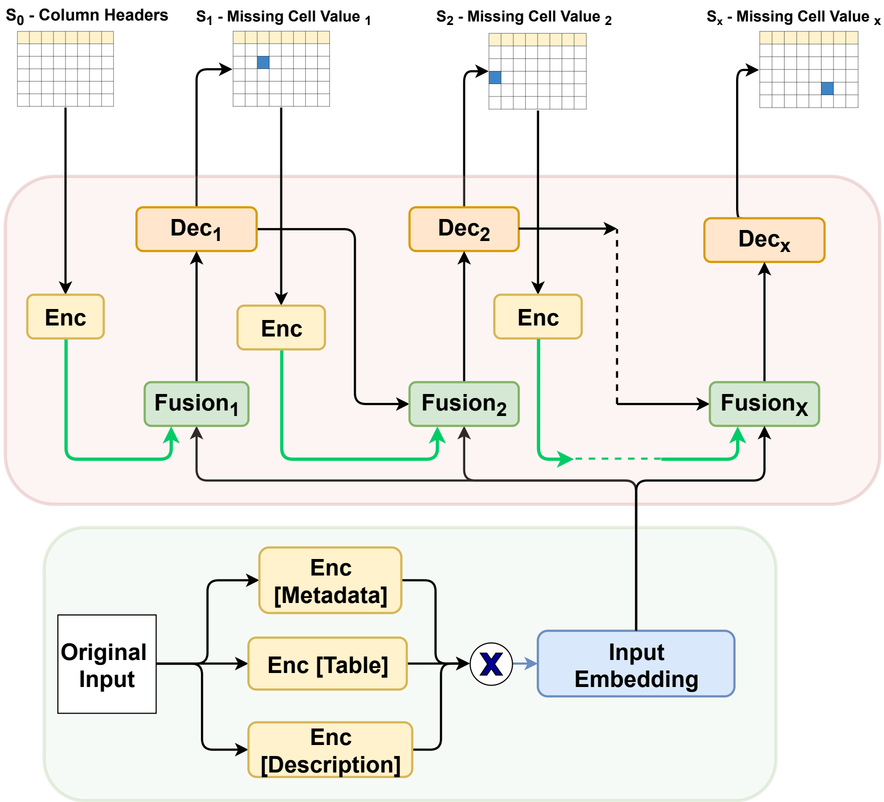

Fig. 4.2. Model Architecture

both encoder and decoder with BERT based weight parameters which is pre-trained on Wikipedia and Books Corpus as BERT has proved to be a versatile model for numerous language tasks. As shown in figure 4.2, an input embedding is obtained using weighted concatenation of all available input encoding. In the described model, stopping criteria is fixed for each data sample which is equal to the number of MCV tokens.

For the first missing cell values S_1, only the column headers from S_0 are passed into the fusion gate for context. Fusion gate captures the relationship between previous missing cells, thus providing better context for the current state. For each step S_i, fusion gate learns weights to be assigned to each incoming element, namely, input embedding, previous output and previous state hidden representation. Similarly, for each step S_i, the decoder Dec_i learns individually for that particular missing cell value i. Note that teacher forcing technique is used to avoid propagation of erroneous outputs from the initial steps. In the early steps while the model is still learning, it may produce incorrect values which are passed forward to generate next missing values which leads to poor performance. To avoid the situation, we pass the ground truth values to the decoder while training. 'MuCeF: Multiple Cell Filler' is build with decoder chain length equal to three. Experimentally we found it to be an ideal length but it can be varied based on the task.

$$InputEmbedding = \sigma(w1 * subtable + w2 * description + w3 * metadata) \tag{4.7}$$

$$Fusion^i = \begin{cases} \sigma(w_4^0 * InputEmbedding + w_5^0 * ColumnHeaders), \text{if i == 0} \\ \\ \sigma(w_4^i * InputEmbedding + w_5^i * PreviousState), \text{otherwise} \end{cases} \tag{4.8}$$

Chapter 5

Results

5.1 Baseline and Model Comparison

In table 5.1, we compare MuCeF with four baselines, namely, TAPAS [24], T5 [32], BERT [33] and sequence to sequence model [6] with one layer LSTM units.

Model	ROUGE-1	ROUGE-2	ROUGE-3	ROUGE-L	METEOR
Seq2seq	15.69	9.49	–	15.16	5.35
BERT	29.61	27.41	4.43	29.50	10.87
T5	32.93	15.57	6.22	31.88	11.96
TAPAS	35.35	18.15	8.23	34.67	14.80
MuCeF	**53.24**	**39.94**	**15.05**	**50.11**	**49.35**

Table 5.1: Baselines and model comparison.

5.2 Error Analysis

In text generation tasks, automatic evaluation metrics are useful to an extent only. Qualitative analysis is done by humans to get more thorough insights. Here, we select a random subset of 100 data samples and analyze it based upon two factors: first, in how many samples the system needs to infer from metadata/ text description along with the table and second, in how many samples the system needs background world knowledge to generate outputs. Results are given in table 5.2. Almost all

Property	**Value**
Requires inference from metadata or text description	98%
Requires background knowledge	14%

Table 5.2: Analysis

data samples requires logical inference from all the three input modalities i.e. text description, metadata and the table. Around 14% of tables in the subset required general world knowledge to generate fluent text, for example, knowing proper sports terms to be used in the text or knowing full dates of historical events.

Error analysis carried out on the subset of 100 samples reveals the following problems in output values. First is that the system favours shorter text specially in case of named entities. For example, generated value is ”Arthur III” while the expected value is ”Arthur III the Justicier (Arzhur III)”, generating ”Strikeforce Challengers: Voe” instead of expected ”Strikeforce Challengers: Voelker vs. Bowling III”. Another kind of error is in alphanumerical values where again the system hallucinates the value or produce a closely related value. For example, generating ”86xx” instead of ”8681”, generating ”162A” instead of ”ZAR162A10”. Apart from that, some recurring text can be seen in output. A main advantage of MuCeF is that it produces exactly the number of sequences that are required unlike standard language models. Few output examples can be seen in table 5.3. Figure 5.1 shows a data example in test set with ground truth compared to output generated by MuCeF and TAPAS.

Ground Truth	**MuCeF Output**	**Best baseline Output**
Peter II the Simple (Pêr II) 1450–1457, Arthur III the Justicier (Arzhur III) 1457–1458, 26 December 1458 Nantes aged 65	Peter II the Simple (Pêr II) 1450 – 1457 Arthur III the Justicier (Arzhur III) 1457 – 1458 26 December 14 58 Nantes aged 65	Arthur III (Born: c. 494 Died: 4 September 1831)
Best Actor Won	Best Act or Won	Best Actor in the Silent Scream
Ralph J. Parker	Ralph J. Parker	Ralph J. Parker (inc.)
2016, Da Nang, Vietnam	2016, Da N ang, Vietnam	2016–16 (2012–2016)

Table 5.3: Comparing ground truth with output from MuCeF and the best baseline

David Storl

Section Title: Achievements
Table Section Text: *None*

Year	Competition	Venue	Position	Event	Notes
Representing Germany					
2007	World Youth Championships	Ostrava, Czech Republic	1st	Shot put (5 kg)	21.40 m
2008	World Junior Championships	Bydgoszcz, Poland	1st	Shot put (6 kg)	21.08 m
2009	European Junior Championships	Novi Sad, Serbia	1st	Shot put (6 kg)	22.40 m
	World Championships	Berlin, Germany	27th (q)	Shot put	19.19 m
2010	World Indoor Championships	Doha, Qatar	7th	Shot put	20.40 m
	European Championships	Barcelona, Spain	4th	Shot put	20.57 m
	European Indoor Championships	Paris, France	2nd	Shot put	20.75 m
		Ostrava, Czech Republic	1st	Shot put	20.45 m
	World Championships	Daegu, South Korea	1st	Shot put	21.78 m
	DÃ©caNation	Nice, France	1st	Shot put	20.30 m
2012	World Indoor Championships	Istanbul, Turkey	2nd	Shot put	21.88 m
	European Championships	Helsinki, Finland	1st	Shot put	21.58 m
	Olympic Games	London, United Kingdom	2nd	Shot put	21.86 m
2013	World Championships	Moscow, Russia	1st	Shot put	21.73 m

Sentence(s)

David Storl topped the podium at the 2011 European U23 Championships with a record of 20.45 m.

Ground Truth	MuCeF	Top Baseline
2011	2011	2011–12 (European U23 Championships)
European U23 Championships	European U23 Championships	

Fig. 5.1. Output Comparison between Ground Truth, MuCeF and TAPAS.

Chapter 6

Conclusion

In this paper, we present a sequence-to-multi-sequence learning approach which we experiment on a modified dataset from Wikipedia for the task of populating missing cell values in a partially filled table. As discussed, the system is flexible for all kind of domains such as audio and NLP as well as it is customizable for type of encoder/decoder the user may like to use. Main challenges in using the model are exposure bias due to teacher forcing method and increased space complexity due to increased length of encoder-decoder chain which can be addressed by changing to optimal number of decoders. Overall, we see an improvement of 15.44 ROUGE score and 34.54 METEOR score than the top baseline. The dataset and code for this work will be publicly released for further research.

Bibliography

[1] K. McKeown, *Text Generation*, ser. Studies in Natural Language Processing. Cambridge University Press, 1985.

[2] S. Wiseman, S. M. Shieber, and A. M. Rush, "Challenges in data-to-document generation," 2017.

[3] A. P. Parikh, X. Wang, S. Gehrmann, M. Faruqui, B. Dhingra, D. Yang, and D. Das, "Totto: A controlled table-to-text generation dataset," 2020.

[4] T. Liu, K. Wang, L. Sha, B. Chang, and Z. Sui, "Table-to-text generation by structure-aware seq2seq learning," *arXiv preprint arXiv:1711.09724*, 2017.

[5] K. Filippova, "Controlled hallucinations:learning to generate faithfully from noisy data," in *Findings of EMNLP 2020*, 2020.

[6] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," 2014.

[7] J. Shi, X. Chang, P. Guo, S. Watanabe, Y. Fujita, J. Xu, B. Xu, and L. Xie, "Sequence to multi-sequence learning via conditional chain mapping for mixture signals," 2020.

[8] M. Johnston, S. Bangalore, G. Vasireddy, A. Stent, P. Ehlen, M. Walker, S. Whittaker, and P. Maloor, "MATCH: An architecture for multimodal dialogue systems," in *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*. Philadelphia, Pennsylvania, USA: Association for Computational Linguistics, Jul. 2002, pp. 376–383. [Online]. Available: https://aclanthology.org/P02-1048

[9] R. Kondadadi, B. Howald, and F. Schilder, "A statistical NLG framework for aggregated planning and realization," in *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*.

Sofia, Bulgaria: Association for Computational Linguistics, Aug. 2013, pp. 1406–1415. [Online]. Available: https://aclanthology.org/P13-1138

[10] W. Lu, H. T. Ng, and W. S. Lee, "Natural language generation with tree conditional random fields," in *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*. Singapore: Association for Computational Linguistics, Aug. 2009, pp. 400–409. [Online]. Available: https://aclanthology.org/D09-1042

[11] A. Stent, R. Prasad, and M. Walker, "Trainable sentence planning for complex information presentation in spoken dialog systems," in *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*, ser. ACL '04. USA: Association for Computational Linguistics, 2004, p. 79–es. [Online]. Available: https://doi.org/10.3115/1218955.1218966

[12] J. Gu, Z. Lu, H. Li, and V. O. Li, "Incorporating copying mechanism in sequence-to-sequence learning," in *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Berlin, Germany: Association for Computational Linguistics, Aug. 2016, pp. 1631–1640. [Online]. Available: https://aclanthology.org/P16-1154

[13] C. Rebuffel, L. Soulier, G. Scoutheeten, and P. Gallinari, "A hierarchical model for data-to-text generation," 2019.

[14] S. Choi, J. in Hwang, H. Noh, and Y. Lee, "May the force be with your copy mechanism: Enhanced supervised-copy method for natural language generation," 2021.

[15] W. Chen, Y. Su, X. Yan, and W. Y. Wang, "KGPT: Knowledge-grounded pre-training for data-to-text generation," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Online: Association for Computational Linguistics, Nov. 2020, pp. 8635–8648. [Online]. Available: https://aclanthology.org/2020.emnlp-main.697

[16] L. F. R. Ribeiro, M. Schmitt, H. Schütze, and I. Gurevych, "Investigating pre-trained language models for graph-to-text generation," 2021.

[17] J. Clive, K. Cao, and M. Rei, "Control prefixes for text generation," 2021.

[18] C. Gardent, A. Shimorina, S. Narayan, and L. Perez-Beltrachini, "Creating training corpora for NLG micro-planners," in *Proceedings of the 55th*

Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). Vancouver, Canada: Association for Computational Linguistics, Jul. 2017, pp. 179–188. [Online]. Available: https://aclanthology.org/P17-1017

[19] J. Novikova, O. Dušek, and V. Rieser, "The e2e dataset: New challenges for end-to-end generation," 2017.

[20] E. REITER and R. DALE, "Building applied natural language generation systems," *Natural Language Engineering*, vol. 3, no. 1, p. 57–87, 1997.

[21] L. Li, C. Ma, Y. Yue, and D. Hu, "Improving encoder by auxiliary supervision tasks for table-to-text generation," 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)*. Online: Association for Computational Linguistics, Aug. 2021, pp. 5979–5989. [Online]. Available: https://aclanthology.org/2021.acl-long.466

[22] C. Gardent, A. Shimorina, S. Narayan, and L. Perez-Beltrachini, "The WebNLG challenge: Generating text from RDF data," in *Proceedings of the 10th International Conference on Natural Language Generation*. Santiago de Compostela, Spain: Association for Computational Linguistics, Sep. 2017, pp. 124–133. [Online]. Available: https://aclanthology.org/W17-3518

[23] P. Yin, G. Neubig, W.-t. Yih, and S. Riedel, "Tabert: Pretraining for joint understanding of textual and tabular data," 2020. [Online]. Available: https://arxiv.org/abs/2005.08314

[24] J. Herzig, P. K. Nowak, T. Müller, F. Piccinno, and J. M. Eisenschlos, "Tapas: Weakly supervised table parsing via pre-training," 2020.

[25] Q. Liu, B. Chen, J. Guo, M. Ziyadi, Z. Lin, W. Chen, and J.-G. Lou, "Tapex: Table pre-training via learning a neural sql executor," 2021. [Online]. Available: https://arxiv.org/abs/2107.07653

[26] M. Kale and A. Rastogi, "Text-to-text pre-training for data-to-text tasks," 2021.

[27] A. Moryossef, Y. Goldberg, and I. Dagan, "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)*. Minneapolis, Minnesota: Association for Computational

Linguistics, Jun. 2019, pp. 2267–2277. [Online]. Available: https://aclanthology.org/N19-1236

[28] C. Zhao, M. Walker, and S. Chaturvedi, "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*. Online: Association for Computational Linguistics, Jul. 2020, pp. 2481–2491. [Online]. Available: https://aclanthology.org/2020.acl-main.224

[29] T. Castro Ferreira, C. van der Lee, E. van Miltenburg, and E. Krahmer, "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)*. Hong Kong, China: Association for Computational Linguistics, Nov. 2019, pp. 552–562. [Online]. Available: https://aclanthology.org/D19-1052

[30] Y. Su, D. Vandyke, S. Wang, Y. Fang, and N. Collier, "Plan-then-generate: Controlled data-to-text generation via planning," 2021.

[31] P. von Platen, "Leveraging pre-trained language model checkpoints for encoder-decoder models." [Online]. Available: https://huggingface.co/blog/warm-starting-encoder-decoder

[32] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu, "Exploring the limits of transfer learning with a unified text-to-text transformer," 2020.

[33] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," 2019.