Table-To-Text generation and pre-training with TabT5

Anonymous ACL submission

Abstract

Encoder-only transformer models have been successfully applied to different table understanding tasks, as in TAPAS (Herzig et al., 2020). A major limitation of these architectures is that they are constrained to classification-like tasks such as cell selection or entailment detection. We present TabT5, an encoder-decoder model that generates natural language text based on tables and textual inputs. TabT5 overcomes the encoder-only limitation by incorporating a decoder component and leverages the input structure with table specific embeddings and pre-training. TabT5 achieves new state-of-the-art results on several domains, including spreadsheet formula prediction with a 15% increase in sequence accuracy, QA with a 2.5% increase in sequence accuracy and data-to-text generation with a 2.5% increase in BLEU.

1 Introduction

Large language models (LLMs) such as BERT (Devlin et al., 2019) or T5 (Raffel et al., 2020) have shown impressive abilities to encode and generate fluent and coherent natural language text (Lan et al., 2020; Gururangan et al., 2020; Conneau et al., 2020). However their representation and generational capabilities are limited when it comes to structured or semi-structured domains like tables. This is mainly due to two reasons: (i) LLMs are only pre-trained on large amount of unstructured data (e.g., documents, news, etc.); (ii) their underlying model architecture lacks a way to fully leverage this structure information.

Yet, structured and semi-structured data is ubiquitous on the web (e.g. web tables, database tables, PDF tables, spreadsheets store rich numerical information and provide concise summaries of data), and widely studied in the academia (Chen et al., 2021; Cheng et al., 2022; Parikh et al., 2020; Wang et al., 2021) and the industry (e.g. formula prediction in Excel[1] and Google Sheets[2], or extracting data from tables in Text-to-Speech Assistants).

Recently, several solutions propose to alleviate aforementioned issue by introducing pre-training or intermediate training strategies for tables. For instance, Herzig et al. (2020) propose to use a Masked Language Model (MLM) as pre-training objective to improve the contextual representation of BERT (Devlin et al., 2019) over table inputs. To train their model, they introduce additional input embeddings that help the model understand the table structure. These pre-training models are designed and evaluated on datasets where the answers contain only table cells or aggregations of multiple cells, and not full sentences. In this paper, we tackle a set of distinct, complex tasks such as question answering and formula prediction that require full generation capabilities.

In particular, our contributions are as follows:

- We present an encoder-decoder based model TabT5 (Table-and-Text-to-Text Transfer Transformer) that can be applied to data-to-text generation tasks by relying on special embeddings of the input structure.
- We introduce different pre-training strategies that leverage web data containing tables.
- We evaluate our approach on three different table and text datasets in English and obtain state-of-the-art performance on several domains.

2 Problem Definition

The objective of our model is to learn a conditional sequence generator $P(y|x)$ where x is endowed with extra two-dimensional structure. To encode said structure, each instance of x is as a variable length sequence of tuples $(u_i, t_i, c_i, r_i)_{i=1}^{N}$ representing the *components* of x. In each component, u_i is a natural language utterance, t_i is the discrete

[1] https://www.microsoft.com/excel
[2] https://www.google.com/sheets/about

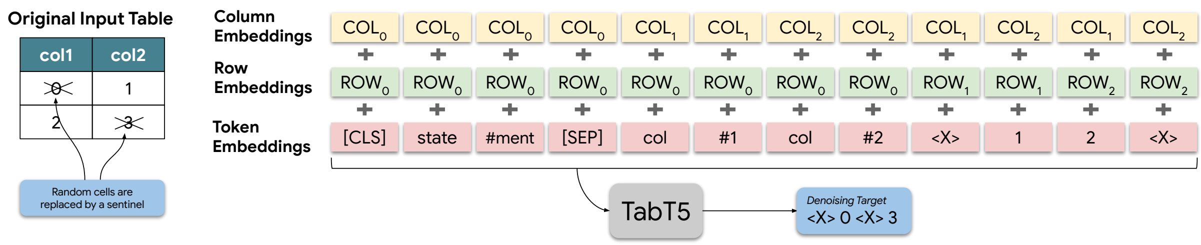

Figure 1: TABT5 linearizes the input table row by row and adds column and row embeddings to encode the 2-dimensional coordinates of each cell. The model is pretrained by randomly replacing 15% of cells by a <X> marker and training the decoder to predict the hidden output in sequence.

type of the component (i.e. could be *Question, Document Title, Table Caption, Table Header, Table Cell*, etc.), and c_i and r_i represent the two dimensional column and row coordinates for this component. This approach is general to represent the information layout in web documents and in particular tables where each table cell and each piece of metadata can map into a single component.

3 Related Work

Table language models. Several works use a common serialization where table contents are linearized row by row (Parikh et al., 2020; Wang et al., 2021; Iida et al., 2021; Eisenschlos et al., 2021; Herzig et al., 2020). Another design choice is to use structural positional encoding, in addition to 1-D encoding, to represent two dimensional information such as the row and column positions of tokens (Herzig et al., 2020; Eisenschlos et al., 2021; Iida et al., 2021; Wang et al., 2021). An alternative is the the use of a structure-aware attention, in contrast to a standard self-attention mechanism, to better leverage the table structure (Mueller et al., 2019; Yang et al., 2022). All of these models are encoder-only. Concurrent with our work Shi et al. (2022) propose a similar method to adapt to T5 to tabular data, however their pretraining approach relies on existing annotated datasets and focuses solely on QA applications.

Table Pre-training. Most pretraining methods follow the Masked Language Modeling (MLM) scheme, where some percentage of input tokens are randomly masked and successively predicted in an encoder only setup (Herzig et al., 2020; Eisenschlos et al., 2021; Yang et al., 2022). Some approaches (Wang et al., 2021; Yin et al., 2020; Iida et al., 2021) apply the masking on a cell-level, where the full contents of a given cell is masked and then predicted. Our work differs in training the encoder and decoder jointly by using a de-noising scheme similar to the one used in T5 (Raffel et al., 2020).

Table QA. Given an input table, the task consists in producing an answer to a natural language question. We focus on WIKISQL (Zhong et al., 2017), and learn an encoder-decoder model with row/column embeddings in the weakly supervised setting without logical forms. Herzig et al. (2020) use a similar approach with a BERT encoder-only model, while Liu et al. (2022) use a BART encoder-decoder model without extra embeddings.

Formula prediction. The task is to predict formula conditioned on headers and other contextual information, without an explicit natural language question. Chen et al. (2021) propose to use a BERT-based architecture to compute an input header and a cell data vector that are fed to a two-step LSTM decoder. The decoder proposes a formula sketch and refines it with cell ranges. Cheng et al. (2022) propose a similar approach where the representation of the target cell output by a table encoder (Wang et al., 2021) is an input to a two-step LSTM-based decoder. Our approach is simpler as a single model is used to solve the task end to end.

Data-to-Text. The task consists in generating a natural language description given structured data input. Parikh et al. (2020) employ an encoder-decoder model where the encoder and decoder are both initialized with BERT (Devlin et al., 2019). Kale and Rastogi (2020) use a T5 model. In both, tables are linearized with row/column separator tokens. Our work differs as we use row/column embeddings, and we employ two pretraining schemes.

4 TABT5 Model

TABT5 uses the T5 pre-trained language model as a baseline architecture.We linearize the table into a sequence of words, split words into word pieces (tokens) and concatenate the question and table tokens to create the input sequence. We include in the model row and column embeddings to encode table structure (Herzig et al., 2020). We add them on top of the token embeddings as model inputs and optimize them during training (Figure 1). The target sequence is a free-form answer. This can be an answer to a question for question-answering tasks, a table summary when no question is specified or a formula for the formula prediction tasks.

5 Pre-training

As a starting point, we use publicly available T5 checkpoints released by Raffel et al. (2020). Next, we pre-train TabT5 on Wikipedia tables. We use the pre-training dataset proposed by Herzig et al. (2020) which contains 6.2M tables (3.3M of class Infobox[3] and 2.9M of class WikiTable). We also extract related passages that caption the table. We define two pre-training strategies described below.

5.1 Denoising

We design a denoising strategy for table-like data, following the method used in T5 (Raffel et al., 2020), by training the model to predict a target sequence containing the missing or corrupted tokens in the input table. The target consists of all of the dropped-out spans of tokens, delimited by the sentinel token used in the input sequence (Figure 1). We replace 15% of table cells and columns in the input with a mask token[4]. This helps the model capture relationships between the neighbouring cells and between the related text.

5.2 ToTTification

We define another pre-training strategy using the same Wikipedia tables (Section 5.1) inspired by ToTTo (Parikh et al., 2020), to be used after denoising. For each table, we retrieve the statements that are in the same page as the table or link to the table page. We only keep statements that have an entity (Wikipedia URL, number or date) that matches the table, 4M in total. These statements become our target text. We add the matching entities in those statements as a (comma separated) plain text component of the input to guide the generation.

6 Experiments

6.1 Datasets

WikiSQL (Zhong et al., 2017) is a Table-QA dataset containing 80.654 instances. To create the dataset, crowd workers paraphrase a template-based question into natural language. Two other crowd workers' groups then verify and correct the quality of the proposed paraphrases. We follow the approach of Herzig et al. (2020) and generate the reference answer from the reference SQL provided using our own SQL implementation.

Enron (Chen et al., 2021) is a dataset to evaluate formula prediction task containing over 17K spreadsheets extracted from the Enron email corpus that contains 218.798 instances. It focuses on formula with referenced cells in a rectangular neighbourhood region of the target cell and the headers. We preprocess the data as described Appendix C.

ToTTo (Parikh et al., 2020) is a Table-to-Text dataset containing 120.761 instances. It consists of tables paired with table-grounded sentences as natural language descriptions. Parikh et al. apply several heuristics to sample tables and candidate sentences from Wikipedia pages. They use crowd worker annotators to highlight the corresponding table cells and revise natural language descriptions.

6.2 Results

We discuss the experimental setup in the Appendix B. For ToTTo, we report the results in Table 1. We follow Parikh et al.'s official script to compute BLEU and PARENT as the evaluation metrics. The *Non-Overlap* dev set features examples that are out-of-domain from the training set. For the test set, we provide results from one run as this is a laborious manual process requiring a submission of test files into an external source[6]. Note that Parikh et al. do not provide development set results in their paper and Kale and Rastogi do not provide test set results for the base model. We observe that TabT5 outperforms SOTA models and its performance is improved further by using the ToTTify pre-training. Note that the base model performs slightly better than the large model. We believe that the large model requires more careful hyperparameters tuning to achieve higher results. For WikiSQL and Enron, results are reported in Table 2 and Table 3 respectively. Also, see the Appendix C for additional results on the Enron dataset. We observe that TabT5 significantly improves over SOTA performance for both WikiSQL ($>$ 30% of error reduction in the base variant) and Enron (35% error reduction in the base variant). Note that TabT5 in the base variant (220M parameters) outperforms other models with substantially

[3] https://en.wikipedia.org/wiki/Help:Infobox

[4] Raffel et al. experimentally show that 15% corruption rate works best. We use the same rate for our denoising objective.

[6] The details on submissions for the ToTTo test set can be found in https://github.com/google-research-datasets/ToTTo

	Overall		*Non-Overlap*	
Model – Dev Set	**BLEU**	**PARENT**	**BLEU**	**PARENT**
Kale and Rastogi	47.70	57.10	39.60	52.60
T5-base[5]	47.00 ± 0.43	55.96 ± 0.31	38.50 ± 0.48	51.14 ± 0.33
TABT5-small	47.80 ± 0.26	56.89 ± 0.29	39.30 ± 0.26	51.93 ± 0.35
TABT5-base	49.00 ± 0.07	57.70 ± 0.11	40.90 ± 0.13	53.12 ± 0.18
+TOTTIFY	$\mathbf{49.50 \pm 0.07}$	$\mathbf{57.95 \pm 0.05}$	$\mathbf{41.60 \pm 0.05}$	$\mathbf{53.65 \pm 0.07}$
-DENOISING	47.50 ± 0.43	56.11 ± 0.40	39.00 ± 0.64	51.06 ± 0.51
-EMBEDDINGS	48.60 ± 0.17	57.12 ± 0.23	40.50 ± 0.26	52.71 ± 0.28
TABT5-large	48.50 ± 0.13	56.98 ± 0.25	41.05 ± 0.10	52.95 ± 0.24
Model – Test Set	**BLEU**	**PARENT**	**BLEU**	**PARENT**
Parikh et al.	44.00	52.60	35.10	46.80
T5-base	47.10	56.17	38.70	51.39
TABT5-base	48.80	57.60	40.70	53.20
+TOTTIFY	**49.20**	**57.25**	**41.00**	**52.78**

Table 1: Text generation results for TOTTO on development (dev) and test sets. The *Non-Overlap* set features examples that are out-of-domain from the training set. TABT5 provides improvements over existing approaches and TOTTIFY pretraining provides additional gains.

higher number of parameters (e.g. BERT used in Herzig et al. (2020) has 380M parameters and BART_{large} in Liu et al. (2022) $\sim$ 418M). Additionally, TABT5 in the small variant (60M parameters) achieves high accuracy compared to SOTA for the ENRON dataset. When increasing the model size, we observe an increase in performance for both datasets. For WIKISQL the large variant (770M parameters) achieves exceptionally high sequence accuracy of 95% (53% error reduction wrt. to the baseline performance).

Model	**Dev**	**Test**
Herzig et al. (2020)	85.1	83.6
Liu et al. (2022)	89.2	89.5
T5-base	85.29 ± 0.45	84.27 ± 0.39
TABT5-small	90.56 ± 0.15	89.15 ± 0.10
TABT5-base	92.55 ± 0.23	91.45 ± 0.21
+TOTTIFY	91.34 ± 0.17	90.06 ± 0.15
-DENOISING	88.87 ± 0.31	87.51 ± 0.19
-EMBEDDINGS	85.51 ± 0.23	84.39 ± 0.13
TABT5-large	$\mathbf{94.92 \pm 0.04}$	$\mathbf{93.61 \pm 0.09}$

Table 2: Table-QA results on WIKISQL in the weakly supervised setting without logical forms. TABT5 provides gains over existing approaches even in a small model variant. The large model gives the best results.

Ablation We perform the ablation study only on the base variant due to computational costs. For each experiment, we report two ablations runs: (i) -DENOISING indicates that we remove the denoising pre-training , and (ii) -EMBEDDINGS indicates runs without row and column embeddings. We observe that the performance of TABT5 deteriorates when removing either denoising or embeddings. This show they are crucial for all tasks. We also show results for the TOTTIfication, which improves the performance for TOTTO but it is detrimental for the other tasks, compared to the denoising method.

Model	**Top-1**
Chen et al.	42.51
Cheng et al.	56.30
T5-base	69.40 ± 0.33
TABT5-small	71.33 ± 0.24
TABT5-base	71.61 ± 0.27
+TOTTIFY	71.18 ± 0.22
-DENOISING	70.47 ± 0.28
-EMBEDDINGS	70.07 ± 0.36
TABT5-large	$\mathbf{71.79 \pm 0.20}$

Table 3: Formula prediction results on ENRON. The T5-base baseline brings substantial improvements over existing approaches. TABT5 provides further gains, with the large model variant obtaining the best results.

Error analysis We manually annotate a random sample of 80 errors made by TABT5. We find that 55% are paraphrases and 72.5% overall are acceptable (correct content with some details missing). We classify the remaining errors into grammatical errors, hallucinations and wrong answers (see Appendix D). The results suggest a need for better metrics for the data-to-text generation tasks that capture the similarities.

7 Conclusions and Discussion

We introduced TABT5 a new T5-based encoder-decoder model that achieves new SOTA results on spreadsheet formula prediction, question answering and data-to-text generation. In future work, we plan to use larger and task specific datasets for pre-training (e.g. scrape tables from Web, sheets).

[6]The gap between T5-base and Kale and Rastogi comes from using distinct versions of T5. We reproduced their results using v1.0. We use v1.1 with the same set of hyperparameters.

Ethical Considerations

As is true for existing works on generative architectures based on large language models, there are potential risks and harms associated with using the output for downstream applications (Bender and Koller, 2020; Brown et al., 2020). Beyond the original pre-trained checkpoint from T5, we also used tables from Wikipedia for intermediate pre-training, which may contain additional undesirable biases.

References

Emily M. Bender and Alexander Koller. 2020. Climbing towards NLU: On meaning, form, and understanding in the age of data. In *Proceedings of the Association for Computational Linguistics*, pages 5185–5198, Online. Association for Computational Linguistics.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Proceedings of Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Xinyun Chen, Petros Maniatis, Rishabh Singh, Charles Sutton, Hanjun Dai, Max Lin, and Denny Zhou. 2021. Spreadsheetcoder: Formula prediction from semi-structured context. In *Proceedings of the International Conference of Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 1661–1672. PMLR.

Zhoujun Cheng, Haoyu Dong, Fan Cheng, Ran Jia, Pengfei Wu, Shi Han, and Dongmei Zhang. 2022. Fortap: Using formulae for numerical-reasoning-aware table pretraining. In *Proceedings of the Association for Computational Linguistics*.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the Association for Computational Linguistics*, pages 8440–8451, Online. Association for Computational Linguistics.

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

Julian Eisenschlos, Maharshi Gor, Thomas Müller, and William Cohen. 2021. MATE: Multi-view attention for table transformer efficiency. In *Proceedings of Empirical Methods in Natural Language Processing*, pages 7606–7619, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A. Smith. 2020. Don't stop pretraining: Adapt language models to domains and tasks. In *Proceedings of the Association for Computational Linguistics*, pages 8342–8360, Online. Association for Computational Linguistics.

Jonathan Herzig, Pawel Krzysztof Nowak, Thomas Müller, Francesco Piccinno, and Julian Eisenschlos. 2020. TaPas: Weakly supervised table parsing via pre-training. In *Proceedings of the Association for Computational Linguistics*, pages 4320–4333, Online. Association for Computational Linguistics.

Hiroshi Iida, Dung Thai, Varun Manjunatha, and Mohit Iyyer. 2021. TABBIE: Pretrained representations of tabular data. In *Conference of the North American Chapter of the Association for Computational Linguistics*, pages 3446–3456, Online. Association for Computational Linguistics.

Mihir Kale and Abhinav Rastogi. 2020. Text-to-text pre-training for data-to-text tasks. In *International Conference on Natural Language Generation*, pages 97–102, Dublin, Ireland. Association for Computational Linguistics.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. ALBERT: A lite BERT for self-supervised learning of language representations. In *Proceedings of the International Conference on Learning Representations*.

Qian Liu, Bei Chen, Jiaqi Guo, Morteza Ziyadi, Zeqi Lin, Weizhu Chen, and Jian-Guang Lou. 2022. TAPEX: Table pre-training via learning a neural SQL executor. In *International Conference on Learning Representations*.

Thomas Mueller, Francesco Piccinno, Peter Shaw, Massimo Nicosia, and Yasemin Altun. 2019. Answering conversational questions on structured data without logical forms. In *Proceedings of Empirical Methods in Natural Language Processing*, pages 5902–5910, Hong Kong, China. Association for Computational Linguistics.

Ankur Parikh, Xuezhi Wang, Sebastian Gehrmann, Manaal Faruqui, Bhuwan Dhingra, Diyi Yang, and Dipanjan Das. 2020. ToTTo: A controlled table-to-text generation dataset. In *Proceedings of Empirical Methods in Natural Language Processing*, pages

1173–1186, Online. Association for Computational Linguistics.

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.

Peng Shi, Patrick Ng, Feng Nan, Henghui Zhu, Jun Wang, Jiarong Jiang, Alexander Hanbo Li, Rishav Chakravarti, Donald Weidner, Bing Xiang, and Zhiguo Wang. 2022. Generation-focused table-based intermediate pre-training for free-form question answering. In *Association for the Advancement of Artificial Intelligence*. AAAI Press.

Zhiruo Wang, Haoyu Dong, Ran Jia, Jia Li, Zhiyi Fu, Shi Han, and Dongmei Zhang. 2021. TUTA: Tree-based transformers for generally structured table pre-training. In *Knowledge Discovery and Data Mining*, KDD '21, page 1780–1790, New York, NY, USA. Association for Computing Machinery.

Jingfeng Yang, Aditya Gupta, Shyam Upadhyay, Luheng He, Rahul Goel, and Shachi Paul. 2022. TableFormer: Robust transformer modeling for table-text encoding. In *Proceedings of the Association for Computational Linguistics*.

Pengcheng Yin, Graham Neubig, Wen-tau Yih, and Sebastian Riedel. 2020. TaBERT: Pretraining for joint understanding of textual and tabular data. In *Proceedings of the Association for Computational Linguistics*, pages 8413–8426, Online. Association for Computational Linguistics.

Victor Zhong, Caiming Xiong, and Richard Socher. 2017. Seq2SQL: Generating structured queries from natural language using reinforcement learning. *CoRR*, abs/1709.00103.

A Appendix A. Hyperparameters Selection

We run denoising pre-training for 1M steps and ToTTify pre-training for 100k steps on top of denoising. We set each fine-tuning task for 50k training steps. We run the evaluation on ENRON and WIKISQL using the default T5 hyper-parameters with an input sequence length of 1024 and output 256. For the TOTTO dataset, we follow the approach of Kale and Rastogi (2020) and keep the learning rate constant and equal to 1×10^{-4} , an input and output sequence length is equal to 512 and 256 respectively, and batch size is 256. Additionally, we observe that TABT5 in the small and base variants overfit quickly. Thus, we decide to increase the dropout rate to 0.2 when using pre-training.

B Appendix B. Experimental Setup

We apply the standard T5 tokenizer and start pre-training from publicly available T5 checkpoints. Row and column embeddings are randomly initialized. We run pre-training and fine-tuning on a setup of 16 Cloud TPU v3 cores with maximum sequence length of 1024. Pre-training takes around 3, 8 and 13 days for small, base and large models. Fine-tuning takes around $2-3$ hours for each task. For each dataset, we run five independent runs and report median and standard deviation.

C Appendix C. ENRON results.

In this section, we present the results on the ENRON dataset that contains all original data (i.e. all formulas in the tables). We find these results interesting as the ENRON contains real data collected by the company. Thus, we believe this scenario is realistic. We present the results in Table 4. We observe that our results are extremely high because in ENRON dataset over 70% of tables contain a target formula in the input table. Following the previous approaches, we make the task harder. In the experimental section of the paper, we preprocess the datasety by removing all formulas from the input table cells. Additionally, we remove examples containing (i) erroneous formulas, and (ii) ranges from different tables in both input tables and target formulas.

Model	Top-1
T5-base	93.05 ± 0.98
TABT5-small	95.39 ± 0.17
TABT5-base	$\mathbf{95.59 \pm 0.08}$
+TOTTIFY	95.50 ± 0.05
-DENOISING	93.92 ± 0.16
-EMBEDDINGS	95.00 ± 0.16

Table 4: Formula prediction results on ENRON. In this experiment, the model has to produce the target formula having access to the formula used in the surrounding cells. Results are higher wrt. to Table 3 as the model is allowed to “copy” already used formulae or part of them.

D Appendix D. Error analysis

We manually annotate 80 errors made by the TABT5 and classify them in Figure 2. 35% of the TABT5 output are exactly the same as T5’s output where 55% are correct (paraphrases) and 72.5% acceptable answers.

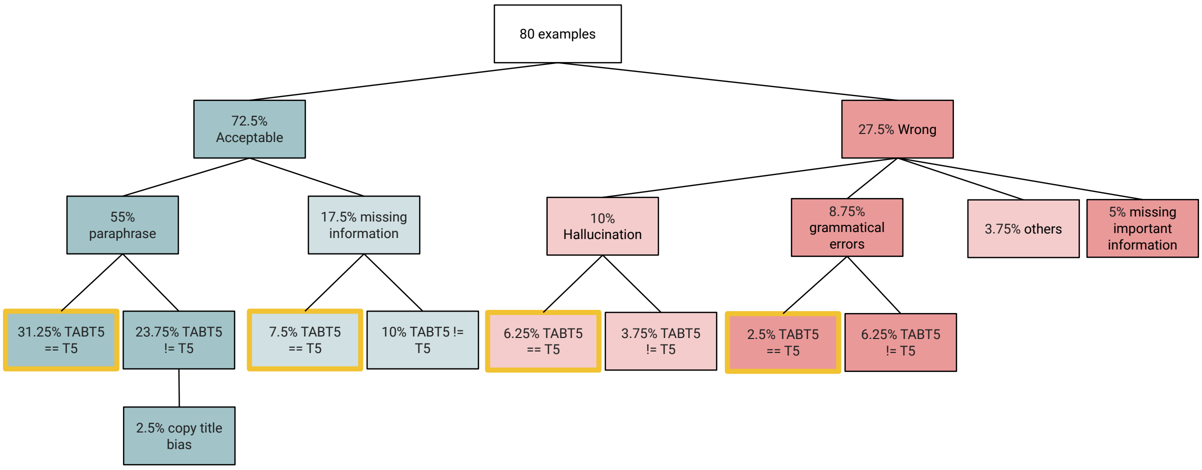

Figure 2: We manually annotate 80 errors made by TABT5. We find that 55% of predictions are paraphrases and 72.5% are acceptable. The classification of error types is given in Table 5.

.

Error type	**Definition**	**Example**
Paraphrase	Express the same meaning as the ground truth using either synonyms or the exact words in a different order.	TABT5 output: Ina 2016, Alma Jodorowsky played Evelyn in Kids in Love. Ground Truth: Alma Jodorowsky had the role of Evelyn in 2016 film Kids in Love.
Acceptable missing information	The content is correct, but it is missing some details that do not affect the answer's meaning.	TABT5 output: The 500 Questions was aired in Germany on RTL from July 4 to August 14, hosted by Günther Jauch. (year is missing) Ground Truth: In 2016, RTL television aired 500-DQA in germany and was hosted by Gunther Jauch
Wrong missing important information	The content is missing some details that affect the meaning of the answer or are essential for understanding the answer.	TABT5 output: Putney railway station is in the Wandsworth borough and is in Zone 2. (missing zone 3 could be important information) Ground Truth: Putney railway station serves Putney in the London borough of Wandsworth in southwest London and in zones 2 and 3
Hallucination	Intrinsic – The generated output contradicts the source content – or Extrinsic – The generated output cannot be verified from the source content –	TABT5 output: In 1924, William Glackens received the Temple Gold Medal for his work "Natural form". (We cannot verify the work's name) Ground Truth: William Glackens won the 1924 award from Temple Gold Medal Nude.
Grammatical errors	The sentence is grammatically incorrect	TABT5 output: As of the census of 2000, there were 42,695 people residing in the Watauga County. Ground Truth: As of the census of 2000, there were 42,695 people residing in Watauga county.
Wrong	Other errors such as: wrong aggregation (counts, sums, etc), swapped arguments that change the meaning of the sentence.	
TABT5 == T5	T5 and TABT5 have the same output –exact match–.	

Table 5: Error types definition and examples.