SciGen: a Dataset for Reasoning-Aware Text Generation from Scientific Tables

Nafise Sadat Moosavi[1], **Andreas Rücklé**[1,*], **Dan Roth**[2], **Iryna Gurevych**[1]

[1]Ubiquitous Knowledge Processing Lab (UKP Lab)
Department of Computer Science, Technical University of Darmstadt
https://www.ukp.tu-darmstadt.de
[2]Department of Computer and Information Science, UPenn
https://www.seas.upenn.edu/~danroth

A Upper-Bound Estimation

To estimate an upper bound for the performance of generation models on our dataset based on automatic metrics, we randomly select 50 tables from expert annotations. We provide these tables to a new expert annotator and ask them to describe the tables in their own words without looking at the gold descriptions or the result sections of the corresponding articles.[2] Table 1 shows the scores of these 50 annotated tables compared to their corresponding gold annotations. The high-value range for all the metrics shows that automatic metrics can acknowledge the correctness of accurate descriptions, i.e., those written by an expert. However, as we see in § 5.1, they cannot discriminate against imperfect descriptions.

BLEU	METEOR	MoverS	BertS	BLEURT
66.73	0.614	0.98	0.99	0.95

Table 1: The automatic evaluation scores for 50 table-descriptions in which an expert has written descriptions based on table contents and without looking at the gold descriptions.

B Impact of Table Captions

Table 2 shows the impact of captions on automatic evaluation metrics. The *caption* row shows the results when the caption is considered as the description, i.e., evaluating captions compared to gold descriptions. The *BART* and *T5-large* rows show the result of these two models in the few-shot setting where the captions of the tables were excluded from the input data.

C Example outputs from all baselines

Table 3 provides the output of all the examined baselines for the table in Figure 2.

D Examples from expert vs. automatically extracted annotations

Below we present a set of examples for comparing expert and automatically extracted description:

[*]Contributions made prior to joining Amazon.

[2]They had access to the background sections of the article in case the information of the table itself is not enough to describe them.

35th Conference on Neural Information Processing Systems (NeurIPS 2021) Track on Datasets and Benchmarks.

Model	**BLEU**	**METEOR**	**MoverS**	**BertS**	**BLEURT**
caption	1.68	0.11	0.12	0.84	-0.98
BART	2.39	0.17	0.08	0.83	-0.72
T5-large	1.50	0.12	0.06	0.81	-0.98

Table 2: Evaluating the impact of table captions. The reported results are equivalent to the reported few-shot results in Table 6 but with the difference that the table captions are excluded from the input. The caption rows show the values of evaluation metrics when the caption is considered as the description.

Setting	**Model**	**Description**
Gold		For ellipsis, both models improve substantially over the baseline (by 19-51 percentage points), with concat stronger for inflection tasks and CADec stronger for VPellipsis.
Zero	**BART**	tied <c >66 . 6 <r ><bold] [empty] s - hier - to - 2 . 2 <c] 76 . 4 <cap >table 8 : accuracy on ellipsis test set . ellipsis (infl .) <c : [bold] ellipis (vp) <c >: [bold] [empty] [cap] [cap] <c . tied [c >53 . 0 <r >. <c >. s - hier - to - 2 <c >65 . 4 [c : [empty] ; [cap] ; <cap ; [cap ; <cap]] <r . <b . c . d . t . t ><c ; [c] : [bold] ; [cap ;] [cap] ; <cap ; [cap] ;] [cap ; <cap] ; »><cap >[cap] [cap] [c] [cap]] [cap]] [cap] [cap]] <cap]] </ cap] <cap] </ cap ><cap »<cap » </ cap »[cap »| cap]]	cap	[cap	<cap	] [cap		cap]	cap [c] [c] ; [c ; [c]] [c]	[c]] [c >||
	T5-base	c >[bold] 76 . 2 c >76 . 6 r >c >s - hier - to - 2 . tied c >66 . 4 c >65 . 6 r >c >s - hier - to - 2 . tied c >66 . 4 c >65 . 6 r >c >[bold] 80 . 0 cap >table 8 : accuracy on ellipsis test set .
	T5-large	<extra$_i d_0 > [bold]ellipsis(vp)c > [bold]ellipsis(infl.)c > [bold]ellipsis(vp)r > c > [empty] c >[bold] ellipsis (vp) c >[empty] c >[bold] ellipsis (vp) r >c >
Few	**BART**	table 8 shows the performance of our model on the ellipsis test set. our concat model outperforms all the other models that do not use concat embeddings.
	T5-base	<extra_id_0 >c >[bold] ellipsis (infl .) c >[s - hier - to - 2 . tied] c >(s - hier - to - 2 . tied) c >s - hier - to - 2 . tied c >(s - hier - to - 2 . tied) c >80 % accuracy on ellipsis test set .
	T5-large	<extra_id_0 >vp) and concatenation (s - hier - to - 2 . tied , infl .) , respectively , are shown in table 8 . as expected , concatenation improves the accuracy on the infl . ellipsis test set .
Medium	**BART**	the results show that both concat and cadec significantly outperform the baseline and s-hier-to-2.tied on the ellipsis test set. the concat model is better than the baseline on both inflected and verb-inflated forms, while cadec is better on the vp version. on the inflated version, cadec outperforms the baseline by a large margin and is within 3 points of the best model. the model with the best accuracy on the verb portion of the test set (80.0%) is very close to the best performance on the noun portion (75.6%). the results indicate that cadec has better generalizability than the other models.
	T5-base	<extra_id_0 >the s - hier - to - 2 . tied model outperforms the baseline on both infl . and vp ellipsis datasets . cadec outperforms s - hier - to - 2 . tied on both infl . and vp ellipsis datasets . cadec achieves the best performance on both infl . and vp ellipsis datasets .
	T5-large	can see that both concatenation and s - hier - to - 2 . tied approaches are significantly better than the baseline . cadec outperforms s - hier - to - 2 . tied on both inflectional and verbal ellipsis and is comparable to the baseline on verb ellipsis .
Large	**BART**	the concatenation model concatenates the output of the concatenated encoder and decoder, and the s-hier-to-2.tied model performs slightly better than concat on both ellipsis (infl.) and vice versa. cadec outperforms the other two models by a large margin, achieving 80.0% accuracy on the vp ellipsoids, which is the new state of the art.
	T5-base	cadec outperforms the baseline and s - hier - to - 2 . tied by a large margin . cadec also outperforms s - hier - to - 2 . tied by a large margin .
	T5-large	concatenated embeddings outperform the s - hier - to - 2 . tied baseline by 6 % on infrequent words , and by 5 % on vp ellipsis . the concatenated embeddings outperform the s - hier - to - 2 . tied baseline by 10 % on infrequent words , and by 5 % on vp ellipsis .

Table 3: Sample outputs from the examined models in Table 6 on the corresponding table from Figure 2.

Expert: (Jeong et al., 2018) Table 1 shows the throughput of training the TreeRNN model using these three datasets. For all batch sizes, the training throughput on the balanced dataset is the highest, while the throughput on the linear dataset is the lowest. As a result, our implementation can train input data of balanced trees with greater throughput than input data of unbalanced trees. Another interesting fact in Table 1 is that the training throughput on the linear dataset scales better than the throughput on the balanced dataset, as the batch size increases. On the contrary, for the linear dataset, the recursive implementation fails to efficiently make use of CPU resources and thus the performance gain provided by increasing the batch size is relatively high.

Automatic: (Jeong et al., 2018) Table 1 shows the throughput of training the TreeRNN model using these three datasets. For all batch sizes, the training throughput on the balanced dataset is the highest,

while the throughput on the linear dataset is the lowest. This trend occurs because the maximum possible execution concurrency of a tree is affected by the balancedness of the tree. A full binary tree of N cells can be processed concurrently with at most N+12 threads, because all N+12 leaf nodes are mutually independent. On the other hand, an extremely unbalanced binary tree can be processed with only one or two threads at a time due to the linearity of the tree. As a result, our implementation can train input data of balanced trees with greater throughput than input data of unbalanced trees. Resource Utilization. Another interesting fact in Table 1 is that the training throughput on the linear dataset scales better than the throughput on the balanced dataset, as the batch size increases. For the balanced dataset, the recursive implementation efficiently utilizes many threads to process the data even at a small batch size of 1, and thus increasing the batch size leads to a relatively small speed boost. On the contrary, for the linear dataset, the recursive implementation fails to efficiently make use of CPU resources and thus the performance gain provided by increasing the batch size is relatively high.

Expert: (Vemulapalli and Agarwala, 2019) Table 3 shows the triplet prediction accuracy of median rater, FECNet-16d and AFFNet-CL-P for each triplet type in the FEC test set. [CONTINUE] the performance is best (85.1%) for two-class triplets, and is lowest (77.1%) for one-class triplets.

Automatic: (Vemulapalli and Agarwala, 2019) Table 3 shows the triplet prediction accuracy of median rater, FECNet-16d and AFFNet-CL-P for each triplet type in the FEC test set. As expected, the performance is best (85.1%) for two-class triplets, which are relatively the easiest ones, and is lowest (77.1%) for one-class triplets, which are relatively the most difficult ones.

References

Eunji Jeong, Joo Seong Jeong, Soojeong Kim, Gyeong-In Yu, and Byung-Gon Chun. 2018. Improving the expressiveness of deep learning frameworks with recursion. In *Proceedings of the Thirteenth EuroSys Conference*, pages 1–13.

Raviteja Vemulapalli and Aseem Agarwala. 2019. A compact embedding for facial expression similarity. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5683–5692.