A How many nuggets in TLDRs?

# categories	0	1	2	3
TLDR-Auth	2.6%	10.5%	26.3%	34.2%
TLDR-PR	0.0%	9.2%	30.3%	31.6%
# categories	**4**	**5**	**6**	
TLDR-Auth	18.4%	7.9%	0.0%	
TLDR-PR	26.3%	2.6%	0.0%	

Table 8: Number of categories represented in a TLDR

B Breakdown of venues in SciTLDR?

Venue	Proportion
ICLR	85.2%
NeurIPS/NIPS	5.8%
OpenReview	2.1%
ICML	2.0%
ICAPS	1.8%
other	3.1%

Table 9: Breakdown of venues represented by papers in SciTLDR

C Background knowledge for TLDRs

What a paper's TLDR looks like or what information it should include is subjective and follows (community-specific) commonsense rather than any formally-defined procedure. Since TLDRs are inherently ultra-short, they are not necessarily self-contained statements, and understanding them requires background expertise within their respective scientific domain. Therefore, when designing SciTLDR, we assume readers have sufficient background knowledge to follow a general research topic in a given domain. This eliminates the need for TLDRs to include explanations or clarifications of common domain-specific terms (e.g., "bounds," "LSTM," or "teacher").

D Additional model training details

PACSUM The default hyperparameters are beta and lambda1 set to 0. We did some initial tuning of the hyperparameters using the provided tuning code, which performs a search over 10 beta values and 10 lambda1 values. This did not result in a significant difference in performance. PACSUM had a total runtime of 12 minutes on abstracts and 6.5 hours on AIC. We used the released code by authors.[19]

BertSumExt We trained with a batch size of 1 sentence per batch and for 5,000 total steps for a total training time of 30 min. We use a learning rate of 2e-3 and a dropout rate of 0.1, which are the reported parameters used for XSUM. BERTSumExt also requires a max token length for initializing position embeddings. For the abstract-only setting, we use the default number of max tokens 512, which fits the full length of all of abstracts in SciTLDR. For AIC, we first attempted 3 different truncation lengths – 1024 (double the max tokens for abstracts), 1500 (90th percentile length), and 1800 (95th percentile length) tokens. We found that truncation at 1500 performs best on AIC. We used the released code by authors.[20]

MatchSum We trained MatchSum with a batch size of 32, learning rate of 2e-5 with a linear warmup and decay scheduler, and trained the model for 15 epochs. We chose the best checkpoint based on linear combination of Rouge-1, Rouge-2 and Rouge-L. We manually tuned hyperparameters – For learning rate, we tried 2e-5 and 3e-5 and for number of epochs, we tried 5, 15, and 20. For AIC, as MatchSum requires few salient sentences as input for candidate generation, we used BertSumExt to score sentences and chose the top 7 ones as input to MatchSum. This is according to instructions by authors[21]. Instead of training the model from scratch we used the authors released checkpoint based on the CNN/DM dataset. This resulted in about 1 Rouge-1 point improvement.

BART For BART and $\text{BART}_{\text{XSUM}}$ finetuning experiments, we train all the models for 500 steps with 20% warm-up for an approximate training time of 45 minutes. This is equivalent to 5 epochs, though we initially allowed BART to train for up to 20 epochs and found that the model quickly overfits to the training set (as evidenced by poor performance on the dev set).

Through manual tuning, we achieved the best results by reducing the training time. Also in manual tuning, we first ran the experiments on four learning rates, 2e-5, 3e-5, 4e-5, and 5e-5 and controlled for all other hyperparameters. We then tested three different seeds, again controlling for all other parameters. Finally, we tested two batch sizes, 2048 tokens per batch and 1024 tokens per batch.

[19] https://github.com/mswellhao/PacSum

[20] https://github.com/nlpyang/PreSumm

[21] https://github.com/maszhongming/MatchSum

CATTS In the abstract-only setting, we train CATTS for 11,000 total steps for a total training time of 2.5 hours. For AIC, we train CATTS for 45,000 total steps for a total training time of 10 hours. This also equivalent to 5 epochs of training. We do not perform tuning on the training hyperparameters for CATTS, instead opting to use the same parameters as the baseline BART models.

E Mean ROUGE test results

	Abstract-only			AIC		
Method	R1	R2	RL	R1	R2	RL
BART	31.1	10.7	24.4	30.7	10.6	24.4
$\text{BART}_{\text{XSUM}}$	30.1	10.7	24.1	31.0	10.9	24.7
CATTS	31.5	11.0	24.9	†31.9	†**11.8**	†**25.6**
$\text{CATTS}_{\text{XSUM}}$	†**31.7**	**11.1**	†**25.0**	†**32.1**	†11.6	†25.4

Table 10: Test set results using mean Rouge scores instead of max for abstractive methods. We use † to indicate CATTS variants that significantly ($p < 0.05$) outperform their corresponding BART baseline.

F TLDR-PR annotation instructions

Below are the instructions provided to annotators rewriting peer-review comments.

Task: We want to collect a dataset of short summaries of CS papers, but it's hard to get people to read and write summaries about entire papers. Instead, we collected a dataset of peer reviewer comments, in which many CS researchers have read and written reviews of papers. Often, a reviewer's comments will also include a summary of the paper they've read. Our task is given the title and first 128 words of a reviewer comment about a paper, re-write the summary (if it exists) into a single sentence or an incomplete phrase. Summaries must be no more than one sentence. Most summaries are between 15 and 25 words. The average rewritten summary is 20 words long.

What might be included in your re-write?

1. What subfield is their work in?
2. What problem are they trying to solve?
3. What did the paper do?
4. Why should you care/how is it novel?

What to exclude when re-writing a comment: Not everything in the reviewer comment belongs in the summary. We purposefully leave out:

- Reviewer decisions/opinions (accept, reject, suggestions, etc.)
  - "The paper is well-written and it is quite easy to follow along with the discussion."
- Background information/ previous work
  - "The authors propose a method for learning node representations which, like previous work (e.g. node2vec, DeepWalk), is based on the skip-gram model."
  - "In particular, when node2vec has its restart probability set pretty high, the random walks tend to stay within the local neighborhood (near the starting node)."
- Excessive details about methodology
  - "Whereas node2vec may sample walks that have context windows containing the same node, the proposed method does not as it uses a random permutation of..."

Enter "None" for the summary for the following conditions:

- The comment is entirely the reviewer's opinions about the paper
- The reviewer's summary carries heavy sentiment about the paper
  - "This paper presents a method that is not novel or interesting"
  - This applies when the sentiment is so heavy that you are unable to write a summary.
- If the comment is about a paper that is out of your domain of expertise.