Generating Abstractive Summaries with Finetuned Language Models

Sebastian Gehrmann* Harvard SEAS
Zachary M. Ziegler* Harvard SEAS
Alexander M. Rush Harvard SEAS
{gehrmann@seas,zziegler@g,srush@seas}.harvard.edu

Abstract

Neural abstractive document summarization is commonly approached by models that exhibit a mostly extractive behavior. This behavior is facilitated by a copy-attention which allows models to copy words from a source document. While models in the mostly extractive news summarization domain benefit from this inductive bias, they commonly fail to paraphrase or compress information from the source document. Recent advances in transfer-learning from large pretrained language models give rise to alternative approaches that do not rely on copy-attention and instead learn to generate concise and abstractive summaries. In this paper, as part of the TL;DR challenge, we compare the abstractiveness of summaries from different summarization approaches and show that transfer-learning can be efficiently utilized without any changes to the model architecture. We demonstrate that the approach leads to a higher level of abstraction for a similar performance on the TL;DR challenge tasks, enabling true natural language compression.

1 Introduction

Abstractive summarization, the challenge of generating text that captures the content of a longer document, has been successfully approached by many recent deep learning systems (e.g., Rush et al., 2015; Nallapati et al., 2016). However, the most common testbed for such methods, news summarization, provides mostly extractive reference summaries which reuse long phrases from the source document. This property gave rise to extensions of neural summarization models that extract text from a source document in addition to generating new words (Vinyals et al., 2015; Gu et al., 2016). As a side-effect, many of the abstractive summarization models have an inductive bias to almost always extract text from the source document verbatim instead of paraphrasing it.

To encourage research on models that can generate summaries that are not extractive, Völske et al. (2017) developed the TL;DR corpus which comprises over three million posts and associated user-written summaries from reddit. Because of the social media nature of the dataset, the user-written summaries copy long sequences from the source much less frequently than common news summarization corpora, resulting in a truly abstractive dataset. This dataset offers the opportunity to investigate the performance of common summarization models in an abstractive setting.

In this work, which is part of the TL;DR challenge, we evaluate and analyze a number of common summarization approaches on both the standard news summarization corpus CNN-DM and the TL;DR dataset. We investigate whether the ability to copy from the source document leads to the same learned extractive behavior, even when the target summaries are mostly abstractive. We additionally evaluate whether neural summarization models can take advantage of pretrained language representation to generate more abstractive text. To measure the abstractiveness of generated summaries, we identify a general "abstractiveness" metric and compare the approaches to the ground truth data for both datasets.

Our results demonstrate that the ability to copy leads to improvements in terms of automated performance evaluation even on the TL;DR dataset, even though it leads to a significantly lower level of abstractiveness. Furthermore, we find that *all* models without pretraining exhibit a significantly higher level of extractiveness than the reference summaries, while language model pretraining allows for more abstractive behavior. Overall, these results suggest that standard summarization approaches learn an easier extractive shortcut than true natural language compression, and that this phenomena occurs even in highly abstractive data.

Proceedings of The 12th International Conference on Natural Language Generation, pages 516–522,
Tokyo, Japan, 28 Oct - 1 Nov, 2019. ©2019 Association for Computational Linguistics

2 Problem and Related Work

Throughout this study, we consider the supervised summarization problem, which aims to compress a source document of tokens $x_1, \dots, x_m$ of length m. The aligned summary $y_1, \dots, y_n$ has a length $n \ll m$, and aims to convey a compressed version of the source document.

Sequence-to-sequence models (S2S, Sutskever et al., 2014) are the de-facto standard for neural abstractive summarization (Rush et al., 2015; Nallapati et al., 2016). The development of models that incorporate a copy-attention mechanism for models to copy word from source documents, has further improved the performance (Gu et al., 2016; Vinyals et al., 2015; See et al., 2017).

However, most summarization tasks use data from news domains which have mostly extractive summaries. Among others, See et al. (2017) and Gehrmann et al. (2018) found that models learn to replicate this latent extraction behavior, and that the resulting summaries of copy-attention based models are over 95% extractive. To address this issue, related approaches have used reinforcement learning objectives to prevent the model from reusing longer phrases from the input and to be more concise (Paulus et al., 2017; Chen and Bansal, 2018; Li et al., 2018). However, these methods often suffer from ungrammatical output or much slower training while also requiring task-specific loss functions. To avoid this problem, Kim et al. (2018) and Völske et al. (2017) created reddit-based corpora with more abstractive target summaries that enable the evaluation of supervised models instead.

Since the generation of abstractive summaries requires a powerful representation of language, we investigate the use of transfer learning. Large language models based on the neural Transformer architecture (Vaswani et al., 2017) have shown promising results in language understanding tasks (Houlsby et al., 2019; Devlin et al., 2018; Chronopoulou et al., 2019), but so far have had limited success in generation tasks (Zhang et al., 2019). Most recently, the pseudo-self attention method for fine-tuning language models to generation tasks has been introduced which may allow the application of transfer-learning to abstractive summarization (Ziegler et al., 2019). In this work, we compare this approach to strong baselines that rely on minor modifications of the Transformer (Gehrmann et al., 2018).

3 Methods

3.1 Models

We consider the following models for neural abstractive summarization. All models are sequence-to-sequence models with attention (Bahdanau et al., 2014), but differ in architecture, use of a copy mechanism, and language model pretraining.

LSTM As a baseline we consider a bidirectional LSTM encoder and uni-directional LSTM decoder with attention from Luong et al. (2015).

LSTM+Copy We additionally consider the same LSTM model equipped with the copy attention mechanism from See et al. (2017). At each time step the approach reuses the normal alignment distribution as a distribution over source words to copy. This copy distribution is combined with the standard target vocabulary distribution from the decoder via a binary switch z_t that is predicted at each time step t.

Transformer(+Copy) For the transformer baselines, we replace the LSTM architecture in the encoder and decoder with transformers (Vaswani et al., 2017). As in the LSTM case we consider version with and without the copy mechanism. Similarly to Gehrmann et al. (2018), we randomly select one of the attention heads as the source of the copy distribution and otherwise follow the same procedure as for the LSTM+Copy.

Transformer+Pretrain Pretrained language models lead to significant performance improvements across a wide range of natural language understanding tasks (Devlin et al., 2018). The recently introduced pseudo self attention method (Ziegler et al., 2019) has also demonstrated strong performance across different generation tasks. The pseudo self attention model follows the same architecture as the original transformer, with minor modifications to inject the context information into the decoder while keeping the structure of the decoder similar to that of an unconditional language model. Most importantly, on the decoder side the context-attention block is removed and the self-attention block is modified to use the source information via pseudo self attention. The normal transformer self-attention computation from Vaswani et al. (2017) can be written most generally as

$$\mathrm{SA}(Y) = \mathrm{softmax}\left((YW_q)(YW_k)^\top\right)(YW_v)$$

where $Y \in T \times D$ is the input and $W_k, W_v, W_q \in D \times D'$ are parameters. In comparison, the pseudo self attention computation is

$$\mathrm{PSA}(X, Y) = \mathrm{softmax}\left((YW_q)\begin{bmatrix} XU_k \\ YW_k \end{bmatrix}^\top\right)\begin{bmatrix} XU_v \\ YW_v \end{bmatrix}$$

where $X \in S \times D$ is the output of the transformer encoder representing the source document and $U_k, U_v \in D \times D'$ are additional parameters.

As in Ziegler et al. (2019), We use the "small" GPT-2 (Radford et al., 2019) as a pretrained unidirectional transformer-based language model. All parameters of the decoder, including the input embeddings, self-attention weights W_k, W_v, W_q for each head and layer, feed forward weights, and layer normalization weights are initialized with the weights from the pretrained language model. The rest of the weights, including those that make up the encoder and the context projections U_k, U_v for each head and layer are randomly initialized. The model is then trained end-to-end on the supervised dataset without fixing any parameters.

Compared to a fully randomly initialized model, the pretrained model has a strong inductive bias towards abstractive generation. Whereas the decoder in a randomly initialized model can learn a generative procedure that largely extracts sequences from the source, the pseudo self attention decoder is initialized with a decoder that already generates coherent language. It may thus be easier for the model to learn to use the source as "inspiration" for the generated text, rather than to learn an entirely different extractive generative procedure. Our experiments aim to quantify this intuition.

3.2 Metric

% novel n-grams One metric used in the literature as a proxy for abstractiveness is the percent of n-grams in the summary that are not found in the source document (See et al., 2017). We report this metric for comparison to previous work.

n-gram abstractiveness While % novel n-grams approximately captures the correct trend, it is poorly normalized: consider a source document *The dog runs around. A cat jumps up. The brown horse stands* and the corresponding summary *The dog runs around. The brown horse stands.* The 4-gram novelty score would identify 4-grams such as *around. The brown horse* as novel, yielding a 4-gram novelty score of 60% even though the summary is composed entirely of copied 4-grams (i.e. a true novelty score should measure 0%). To remedy this, we propose an alternate metric denoted "n-gram abstractiveness":

$$\text{n-gram abstractiveness} = 1 - \frac{\text{\# summary words part of n-gram copied}}{\text{total \# summary words}}$$

To calculate this, we first generate the set of n-grams in the source and summary. All words in the summary which are part of n-grams in the intersection of the two sets are counted as "# summary words part of n-gram copied". Since this (normalized) quantity gives an indication of the fraction of the summary that is copied in n-grams from the source, 1 minus this quantity gives an indication for the abstractiveness of the summary at the n-gram level.

4 Experiments

We compare the presented methods on the non-anonymized CNN-DM dataset (Hermann et al., 2015) and the TL;DR challenge dataset (Völske et al., 2017). CNN-DM comprises roughly 290,000 training examples, which are pruned at a maximum length of 400 words. The corpus is highly extractive, as only 14.0% of tokens in the output do not appear in the corresponding input. Even when we ignore all stopwords, only 17.7% of tokens are novel.

The TL;DR challenge dataset is composed of over three million examples, mined from comments across reddit. We apply the same 400 word pruning to the dataset.The corpus exhibits a much more abstractive behavior, as 53.6% of tokens in the target are novel. After excluding stopwords, this number increases to over 71.4%. That means that this dataset requires a much better text-generating model than CNN-DM.

First baseline models trained on the TL;DR data exhibited a problem that is commonly seen in conversational models in that it defaults to the most simple answer. The simplest answers were a combination of *This is not a problem*; *edit: thank*

	CNN-DM			TL;DR		
Model	R1	R2	RL	R1	R2	RL
LSTM	30.8	11.8	28.5	16	4	13
LSTM+Copy	39.0	16.8	35.7	20	5	15
Transformer	**39.9**	**17.8**	36.6	21	**6**	16
Transformer+Copy	**39.9**	17.7	**37.1**	**22**	**6**	**17**
Transformer+Pretrain	30.5	7.2	28.0	**22**	5	**17**

Table 1: The ROUGE results on the CNN-DM test set and the blind TL;DR test set.

you for the gold; and a number of insults. We thus filtered the dataset by excluding examples in which the target included the following phrases in any capitalization and including common misspellings: *I don't know*; *edit:*; *good idea*; *what I am talking about*; *worth it*; *upvote*; *downvote*; *you'll be fine*; *source:*; and ten different profanities. We further excluded all examples in which the target was shorter than 25 characters to bias the model towards longer generated texts. In total, this procedure excluded 516,000 examples.

Consistent with previous work (Paulus et al., 2017; See et al., 2017; Gehrmann et al., 2018), we find that the LSTM baselines are strongly biased towards short and repetitive summaries. To avoid this, we apply the inference-time loss functions suggested by Gehrmann et al. (2018); a coverage penalty, a length penalty, and a mechanism that prevents repetition of trigrams. We additionally set the minimum length for TL;DR to 25 tokens, which we found to works best on the validation set[1] It is not necessary to apply the same mechanisms to the Transformer-based models. For a better comparison, we only set the minimum length of TL;DR to 25.

5 Automated Evaluation

Table 1 presents the ROUGE scores on the test set for each model on the two datasets[2]. For the LSTM, adding the copy mechanism significantly improves the performance on both the CNN-DM and TL;DR datasets across R1, R2, and RL. Despite the added inference-time loss functions, the LSTM models consistently perform worse than the Transformer models. For the Transformer model, adding the copy mechanism yields a nearly identical performance on CNN-DM and slightly improved performance on TL;DR. Thus, even though the TL;DR dataset is inherently abstractive, copy-attention still improves or is at least no worse in terms of empirical performance.

Using the pretrained representations in the form of pseudo self attention without copy-attention hurts performance considerably on CNN-DM, but slightly improves performance on TL;DR. We hypothesize that this effect can be explained by the abstractiveness of the dataset. Since CNN-DM is mostly extractive, it benefits from the extractive approaches. At the same time, the inductive copying bias has only a minor positive effect on the ROUGE score of the abstractive TL;DR dataset and, thus, a more fluent abstractive summary leads to better performance. Note that here, for the sake of simplicity, we do not consider the Transformer with pseudo self attention and a copy mechanism which was reported to give strong performance in Ziegler et al. (2019).

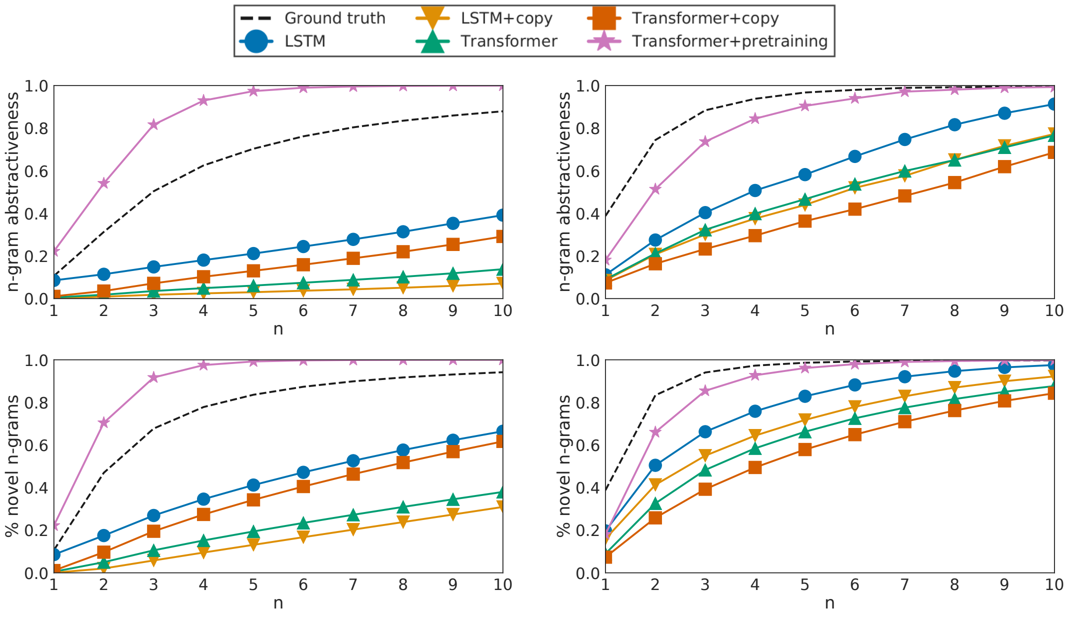
6 Analysis

To validate our hypothesis that pretraining leads to higher abstractiveness, we evaluate the two abstractiveness metrics described in Section 3.2. The results are presented in Figure 1 for all models and the ground truth data on both CNN-DM and TL;DR.

Metric comparison Comparing the overall results for all datasets and models between the proposed n-gram abstractiveness metric and the % novel n-gram metric we find that both metrics present identical trends. The major difference is that the n-gram abstractiveness accounts for the increase in % novel n-grams as n increases, which reduces the noise and leads to a more interpretable

[1] We note that an increased length of generated summaries has been found to increase ROUGE scores which make comparison to other systems with different length outputs challenging (Sun et al., 2019).

[2] The TIRA system (Potthast et al., 2019) used for evaluating the TL;DR task presents scores only with the presented precision.



(a) CNN-DM

(b) TL;DR

Figure 1: The n-gram abstractiveness and % novel n-gram metrics for increasing n show the gap between standard abstractive approaches and the human references. For both corpora, our approach closes this gap.

result. The rest of the analysis will thus focus on the n-gram abstractiveness.

Dataset comparison Comparing the abstractiveness metrics for the reference data between the two datasets provides further evidence that TL;DR is a more abstractive dataset than CNN-DM. While the 4-gram abstractiveness of CNN-DM is only 63%, for example, the 4-gram abstractiveness of TL;DR is 94%. Still, at the higher n-gram levels CNN-DM becomes more than 80% abstractive, suggesting that less than 20% of tokens are part of very long sequences that were copied verbatim.

Randomly-initialized models All randomly initialized models show considerably more extractive behavior than the reference data, for all values of n. This trend exists even for the variants without an explicit copy mechanism and is found in both datasets. This pattern suggests that models trained from scratch may exploit an extractive shortcut which is easier to learn than abstractive data compression.

The addition of a copy mechanism decreases the abstractiveness for all pairs studied expect for the Transformer on CNN-DM. This general trend aligns with the intuition that an explicit copy mechanism allows the model to exploit this easier-to-learn extractive behavior.

Pretraining Compared to the other models, the pseudo self attention pretraining approach leads to a much higher level of abstractiveness. This provides evidence that unlike the randomly initialized models which learn an extractive shortcut, the pretrained model has a strong inductive bias toward abstractive behavior. It is unclear whether this is an artifact of the specific pseudo self attention method or a more general consequence of pretraining for conditional generation.

7 Conclusion

In this paper we study the summarization performance and abstractiveness of summarization models with and without copy attention and pretraining. Combining these two sets of evidence, we find that often the models which perform better are less abstractive, even when the dataset itself is highly abstractive. It is thus challenging to attribute value to abstractiveness when a model is evaluated purely based on its ROUGE score. Our results suggest that if the goal is solely summarization performance, perhaps more extractive models are well suited for this task. Importantly, however, our study emphasizes that despite the performance, we should not be fooled into believing that state-of-the-art summarization models are learning true semantic natural language compression.

Acknowledgements

SG is supported by a Siebel Scholarship. AMR and ZMZ are support by NSF 1845664 and Intel research.

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *CoRR*, abs/1409.0473.

Yen-Chun Chen and Mohit Bansal. 2018. Fast abstractive summarization with reinforce-selected sentence rewriting. *arXiv preprint arXiv:1805.11080*.

Alexandra Chronopoulou, Christos Baziotis, and Alexandros Potamianos. 2019. An Embarrassingly Simple Approach for Transfer Learning from Pretrained Language Models.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding.

Sebastian Gehrmann, Yuntian Deng, and Alexander Rush. 2018. Bottom-up abstractive summarization. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4098–4109.

Jiatao Gu, Zhengdong Lu, Hang Li, and Victor OK Li. 2016. Incorporating copying mechanism in sequence-to-sequence learning. *arXiv preprint arXiv:1603.06393*.

Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. In *Advances in Neural Information Processing Systems*, pages 1693–1701.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin de Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-Efficient Transfer Learning for NLP.

Byeongchang Kim, Hyunwoo Kim, and Gunhee Kim. 2018. Abstractive summarization of reddit posts with multi-level memory networks. *arXiv preprint arXiv:1811.00783*.

Piji Li, Lidong Bing, and Wai Lam. 2018. Actor-critic based training framework for abstractive summarization. *arXiv preprint arXiv:1803.11070*.

Thang Luong, Hieu Pham, and Christopher D Manning. 2015. Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1412–1421.

Ramesh Nallapati, Bowen Zhou, Caglar Gulcehre, Bing Xiang, et al. 2016. Abstractive text summarization using sequence-to-sequence rnns and beyond. *arXiv preprint arXiv:1602.06023*.

Romain Paulus, Caiming Xiong, and Richard Socher. 2017. A deep reinforced model for abstractive summarization. *arXiv preprint arXiv:1705.04304*.

Martin Potthast, Tim Gollub, Matti Wiegmann, and Benno Stein. 2019. TIRA Integrated Research Architecture. In Nicola Ferro and Carol Peters, editors, *Information Retrieval Evaluation in a Changing World - Lessons Learned from 20 Years of CLEF*. Springer.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language Models are Unsupervised Multitask Learners.

Alexander M Rush, Sumit Chopra, and Jason Weston. 2015. A neural attention model for abstractive sentence summarization. *arXiv preprint arXiv:1509.00685*.

Abigail See, Peter J Liu, and Christopher D Manning. 2017. Get to the point: Summarization with pointer-generator networks. *arXiv preprint arXiv:1704.04368*.

Simeng Sun, Ori Shapira, Ido Dagan, and Ani Nenkova. 2019. How to compare summarizers without target length? pitfalls, solutions and re-examination of the neural summarization literature. In *Proceedings of the Workshop on Methods for Optimizing and Evaluating Neural Language Generation*, pages 21–29.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 6000–6010.

Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. 2015. Pointer networks. In *Advances in Neural Information Processing Systems*, pages 2692–2700.

Michael Völske, Martin Potthast, Shahbaz Syed, and Benno Stein. 2017. TL;DR: Mining Reddit to Learn Automatic Summarization. In *EMNLP 2017 Workshop on New Frontiers in Summarization*, pages 59–63. Association for Computational Linguistics.

Haoyu Zhang, Yeyun Gong, Yu Yan, Nan Duan, Jianjun Xu, Ji Wang, Ming Gong, and Ming Zhou. 2019. Pretraining-Based Natural Language Generation for Text Summarization.

Zachary M Ziegler, Luke Melas-Kyriazi, Sebastian Gehrmann, and Alexander M Rush. 2019. Encoder-agnostic adaptation for conditional language generation. *arXiv preprint arXiv:1908.06938*.