Sparsity and Sentence Structure in Encoder-Decoder Attention of Summarization Systems

Potsawee Manakul and **Mark J. F. Gales**
Department of Engineering, University of Cambridge
pm574@cam.ac.uk, mjfg@eng.cam.ac.uk

Abstract

Transformer models have achieved state-of-the-art results in a wide range of NLP tasks including summarization. Training and inference using large transformer models can be computationally expensive. Previous work has focused on one important bottleneck, the quadratic self-attention mechanism in the encoder. Modified encoder architectures such as LED or LoBART use local attention patterns to address this problem for summarization. In contrast, this work focuses on the transformer's encoder-decoder attention mechanism. The cost of this attention becomes more significant in inference or training approaches that require model-generated histories. First, we examine the complexity of the encoder-decoder attention. We demonstrate empirically that there is a sparse sentence structure in document summarization that can be exploited by constraining the attention mechanism to a subset of input sentences, whilst maintaining system performance. Second, we propose a modified architecture that selects the subset of sentences to constrain the encoder-decoder attention. Experiments are carried out on abstractive summarization tasks, including CNN/DailyMail, XSum, Spotify Podcast, and arXiv.[1]

1 Introduction

The Transformer architecture (Vaswani et al., 2017) with large-scale pre-training has become the defacto approach for a wide range of NLP tasks, from classification (Devlin et al., 2019) to seq2seq (Raffel et al., 2020). Training and inference using large transformer models can be computationally expensive because the self-attention's time and memory grow quadratically with sequence length. Hence, there has been significant interest in efficient transformer architectures. A number of approaches have been proposed to tackle the quadratic complexity, and a comprehensive survey on efficient transformers has been compiled in Tay et al. (2020). Most existing approaches are developed for encoder-only architectures. For seq2seq tasks, efficient models such as BigBird (Zaheer et al., 2020) or LED (Beltagy et al., 2020) consist of an efficient encoder with the vanilla decoder. For long-document summarization, this combination has been shown effective because the major bottleneck is the encoder self-attention (Manakul and Gales, 2021). The attention mechanisms in the decoder consist of self-attention and encoder-decoder attention. Techniques such as local attention are applicable to self-attention in both the encoder and decoder, while this work focuses on the encoder-decoder attention.

When humans produce a summary, the information conveyed by each word/part in the summary is likely drawn from some key sentences in the original document. Inspired by this, we hypothesize that if the encoder-decoder attention is constrained dynamically to salient sentences, the computation cost will be reduced. For instance, sentence-level structures for the encoder-decoder attention have been shown effective in the traditional RNN encoder-decoder attention (Cohan et al., 2018; Li et al., 2019; Manakul et al., 2020)

In this work, first, we compare the decoder's cost in the training and inference stages. We study the sparsity of the encoder-decoder attention in a common transformer-based abstractive summarization model. An approximation method to exploit this sparsity is described, and an empirical upper bound performance is given. Second, we propose a modified decoder architecture that can dynamically select salient input sentences to constrain the encoder-decoder attention without having to compute complete attention at inference time. Techniques to train our proposed model are described, and compared to the full attention baseline performance and empirical upper bound.

[1] Our code is available at https://github.com/potsawee/encdec_attn_sparse.

Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, pages 9359–9368
November 7–11, 2021. ©2021 Association for Computational Linguistics

2 Models and Data

Vanilla Transformers. We use BART (Lewis et al., 2020) and local-attention BART (LoBART) (Manakul and Gales, 2021) as our base models. BART's maximum input length is 1024, while that of LoBART is 4096 with attention width of 1024. BART is fine-tuned to CNN/DailyMail and XSum, and LoBART is fine-tuned to Podcast and arXiv.

Data. CNN/DailyMail (Hermann et al., 2015) and XSum (Narayan et al., 2018) are used with BART, while long-document arXiv (Cohan et al., 2018) and Spotify Podcast (Clifton et al., 2020) are used with LoBART. More details about models, data, and training are provided in Appendix A.

3 Attention in the Transformer

Time and memory are dominated by the encoder self-attention, and models such as LoBART adopt local attention in its encoder to mitigate this bottleneck, while keeping the original decoder (Manakul and Gales, 2021). Training is fast because attention is highly parallelizable. However, during inference, the decoder uses its histories, becoming less parallelizable. To understand when the decoder might become a bottleneck, we fix the input length N and measure the computational time as a function of the target length M:

$$\texttt{time} = \bar{c}_1 + \bar{c}_2 M + \bar{c}_3 M^2 \tag{1}$$

in three operating modes: i) Forward+Backward, e.g. at training time; ii) Forward only, e.g. forward-pass where the input to the decoder is provided in advance; iii) Inference, e.g. the decoder using its own back histories as the input.

Through a curve-fitting method, the results in Table 1 show that the relative decoder cost during inference is almost one order of magnitude larger than that during training, e.g. forward+backward or forward only. More details are provided in Appendix B, where we also show that the encoder-decoder attention cost is greater than the decoder self-attention cost. Therefore, this work will focus on the encoder-decoder attention.

3.1 Encoder-Decoder Attention

Let M = the summary length, N = the input length, N_1 = #sentences, and N_2 = the average number of words in a sentence, e.g. $N = N_1 N_2$. The standard encoder-decoder attention in Eq. 2 (scaling factor omitted) where $\mathbf{Q} \in \mathcal{R}^{M \times D}$ and $\mathbf{K}, \mathbf{V} \in \mathcal{R}^{N \times D}$ has the complexity: $\mathcal{O}(MN) = \mathcal{O}(MN_1N_2)$. Note that we fix the representation dimension D, so D is omitted in our complexity notation.

Mode	$\bar{c}_2/\bar{c}_1$ (10^{-3})	$\bar{c}_3/\bar{c}_1$ (10^{-6})
Forward+Backward	1.08	0.17
Forward only	1.14	0.25
Inference	9.96	1.30

Table 1: Empirical computational time as a function of the target length M where $\bar{c}_1, \bar{c}_2, \bar{c}_3$ are the coefficients in Eq. 1. The analysis is based on BART from Wolf et al. (2020) and the input length is 1024.

$$\mathbf{A} = \text{softmax}(\mathbf{Q}\mathbf{K}^T)\mathbf{V} \tag{2}$$

If the attention is concentrated on some r sentences,[2] by selecting appropriate r, the speed of the encoder-decoder attention can be improved by a factor of N_1/r in average. This is equivalent to:

$$\mathbf{A} \approx \hat{\mathbf{A}} = \text{softmax}(\mathbf{Q}\hat{\mathbf{K}}^T)\hat{\mathbf{V}} \tag{3}$$

where $\hat{\mathbf{K}}, \hat{\mathbf{V}} \in \mathcal{R}^{rN_2 \times D}$, resulting in $\mathcal{O}(MrN_2)$.

3.2 Sparsity of Encoder-Decoder Attention

Let the subscript (i, j) denote the position of the j-th word in the i-th input sentence, e.g. $\mathbf{K} = [\underbrace{\mathbf{k}_{1,1}, \mathbf{k}_{1,2}, \mathbf{k}_{1,J_1}}_{\text{sent}1}, ..., \underbrace{\mathbf{k}_{i,1}, \mathbf{k}_{i,J_i}}_{\text{sent}i}, ..., \underbrace{\mathbf{k}_{N_1,1}, \mathbf{k}_{N_1,J_{N_1}}}_{\text{sent}N_1}]$. At inference time, the outputs are generated sequentially: $\mathbf{a}_m$=softmax$(\mathbf{q}_m\mathbf{K}^T)\mathbf{V}$, so r sentences can be determined independently for each $\mathbf{q}_m$. Consider the following sum of attention weights as the saliency at decoding step m of sentence i:[3]

$$\alpha^{\mathsf{s}}_{m,i} = \frac{1}{Z_m} \sum_{j=1}^{J_i} \exp(\mathbf{q}_m \cdot \mathbf{k}_{i,j}) \tag{4}$$

where $Z_m = \sum_{\forall i'} \sum_{\forall j'} \exp(\mathbf{q}_m \cdot \mathbf{k}_{i',j'})$. We then compute $\sum_i \alpha^{\mathsf{s}}_{m,i}$ up to r sentences ranked by $\alpha^{\mathsf{s}}_{m,i}$. The results in Fig. 1a show that r=25 is required to achieve the sum of attention weights at 90%. In addition to the vanilla model, we can fine-tune BART explicitly to make the attention sparse using:

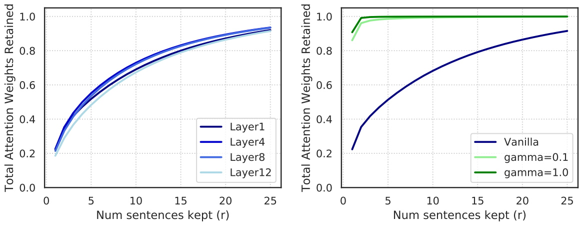
$$\mathcal{L}_{\texttt{A}} = \mathcal{L}_{\texttt{xent}} + \gamma\mathcal{L}_{\texttt{sparse}} \tag{5}$$

where $\mathcal{L}_{\texttt{xent}}$ is the teacher-forced cross entropy loss, $\mathcal{L}_{\texttt{sparse}} = \frac{1}{M}\sum_{m=1}^{M} \text{H}(\boldsymbol{\alpha}^{\mathsf{s}}_m)$, and entropy

[2] Motivated by the observations shown in Appendix E.

[3] We discuss the details of *multi-head* attention on $\alpha^{\mathsf{s}}_{m,i}$ and other operations such as entropy in Appendix A.4

$\text{H}(\boldsymbol{\alpha}^{\mathtt{s}}_m) = -\sum_{i=1}^{N_1} \alpha^{\mathtt{s}}_{m,i} \log \alpha^{\mathtt{s}}_{m,i}$. We show in Fig. 1b that the fine-tuned models (γ=0.1 & γ=1.0) retain close to 100% of attention weights for small r. Subsequently, we investigate how selecting r sentences impacts the summarization performance.



(a) Layers 1,4,8,12

(b) $\mathcal{L}_{\mathtt{A}}$-tuned (Layer1)

Figure 1: The sum of attention weights against the number of retained sentences (r) evaluated on CNNDM.

To obtain an empirical upper bound performance of Eq. 3, for each $\mathbf{q}_m$, we can get *ideal* $\mathbf{k}, \mathbf{v}$ corresponding to the top r sentences ranked by $\alpha^{\mathtt{s}}_{m,i}$:

$$\mathcal{I}^r_m = [(i,j) \text{ s.t. } i \in \text{top-}r(\alpha^{\mathtt{s}}_{m,i})] \qquad (6)$$

$\hat{\mathbf{K}}_m = [\mathbf{k}_{i,j} : (i,j) \in \mathcal{I}^r_m]$, and the same method is applied to obtain $\hat{\mathbf{V}}_m$.

System	r	$\mathcal{I}^r_m$	R1	R2	RL
Vanilla	All	N/A	44.03	20.92	40.99
($\gamma = 0.0$)	5	Ideal	43.94	20.82	40.81
	5	Random	39.06	14.32	36.07
$\gamma = 0.1$	5	Ideal	44.22	21.01	41.19
$\gamma = 1.0$	5	Ideal	43.61	20.46	40.60

Table 2: Sparsity and Selection ($\mathcal{I}^r_m$) on CNNDM.

The results in Table 2 show that:

- For the vanilla model, despite the sum of attention weights being around 50% at r=5 (Fig. 1a), the model is sufficiently sparse, and constraining to r ideal sentences (All $\rightarrow \mathcal{I}^{r,\text{Ideal}}_m$) results in a small performance degradation.
- Forcing for sparsity (Fig.1b) does *not* yield a significant performance improvement; but this forcing also makes the model more sensitive to random selection (results in Appendix C).

Thus, for summarization, there is an observable sparsity, which allows us to reduce the cost of encoder-decoder attention with a minimal degradation. Next, we investigate how to build an efficient form of approximator to obtain salient sentences.

4 Sentence-Level Structure for Encoder-Decoder Attention

In Section 3.1, we use ideal selection $\mathcal{I}^r_m$ (Eq. 6), which requires computing $\alpha^{\mathtt{s}}_{m,i}$ (Eq. 4) using all input words. This process cannot make the decoder more efficient. By exploiting the sentence structure in the document, we propose the following partition for the sentence-level attention score (Eq. 4) to allow a compact approximation:

$$\alpha^{\mathtt{s}}_{m,i} \approx \tilde{\alpha}^{\mathtt{s}}_{m,i} = \text{softmax}\left(f_1(\mathbf{q}_m) \cdot f_2(\mathbf{k}_{i,1}, ..., \mathbf{k}_{i,J_i})\right) \qquad (7)$$

where $\sum_{i=1}^{N_1} \tilde{\alpha}^{\mathtt{s}}_{m,i} = 1.0$. Essentially, we modify the standard encoder-decoder attention such that it performs sentence selection based on $\tilde{\alpha}^{\mathtt{s}}_{m,i}$ (Eq. 7) and computes subset attention $\hat{\mathbf{A}}$ (Eq. 3).

4.1 Complexity of Modified Attention

The modified encoder-decoder attention consists of two components: i) sentence-level attention over N_1 sentences; ii) word-level attention over rN_2 words. Let p denote a unit of matrix multiplication cost and q denote a unit of softmax cost. The costs associated with attention are:

i) Sentence-level (Eq.7): $pMN_1D + qMN_1$

ii) Word-level (Eq.3): $2pMrN_2D + qMrN_2$

The additional cost associated with the sentence-level representation on the encoder side grows with the input length N=N_1N_2. Thus, as opposed to $\mathcal{O}(MN_1N_2)$ in the case of vanilla encoder-decoder attention, the overall complexity of the modified attention is $\mathcal{O}(MN_1 + k_wMrN_2 + k_eN_1N_2)$, where $k_w \approx \frac{2pD+q}{pD+q}$ and k_e depends on the exact form of sentence-level representation computation.

4.2 Model-based Neural Approximator

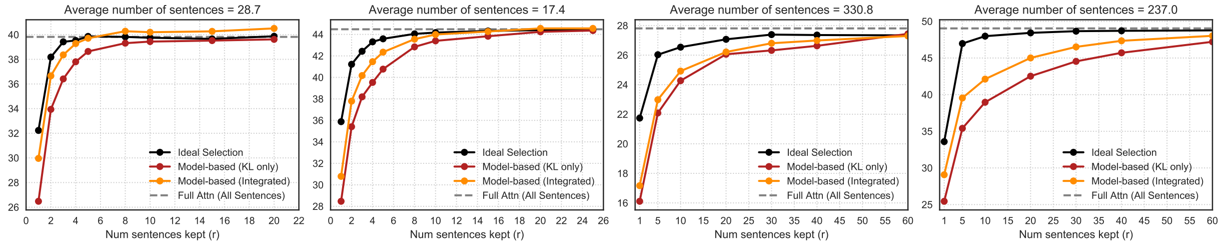
To utilize the simple partition and sentence-level structure in Eq. 7, we use a linear mapping for f_1 and a bidirectional RNN for f_2 as follows:

$$f_1(\mathbf{q}_m) = \mathbf{q}_m\mathbf{W}^{\mathtt{Q}} \qquad (8)$$

$$f_2(\mathbf{k}_{i,1}, ..., \mathbf{k}_{i,J_i}) = \mathbf{y}_i\mathbf{W}^{\mathtt{K}} \qquad (9)$$

$$\mathbf{y}_i = \text{RNN}(\mathbf{k}_{i,1}, ..., \mathbf{k}_{i,J_i}) \qquad (10)$$

As illustrated in Fig. 3, the base transformer model is extended by augmenting two layers: i) sentence-level encoder-decoder attention computing $\tilde{\alpha}^{\mathtt{s}}_{m,i}$ in Eq. 7; ii) sentence encoder computing the sentence-level representation in Eq. 10. The details about model parameters are provided in Appendix A.



(a) BART(1k) & CNNDM (b) BART(1k) & XSum (c) LoBART(4k) & Podcast (d) LoBART(4k) & arXiv

Figure 2: Performance (ROUGE-1) of BART & LoBART. The integrated training is based on $\mathcal{I}_m^{r,\text{Apx}}$.

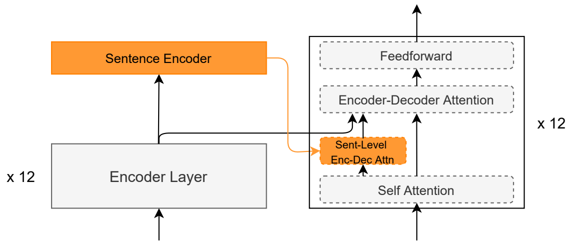


Figure 3: Modified architecture with model-based approximator where the base model is BART/LoBART. Model-based neural approximator is shown in orange.

4.3 KL Loss and Integrated Training

Let $\boldsymbol{\theta}_{\text{dec}}$ denote the original decoder, and $\tilde{\boldsymbol{\theta}}$ denote the neural approximator. We train $\tilde{\boldsymbol{\theta}}$ by minimizing:

$$\mathcal{L}_{\text{KL}} = \frac{1}{M}\sum_{m=1}^{M} \text{KL}\left(\boldsymbol{\alpha}_m^{\text{s}} || \tilde{\boldsymbol{\alpha}}_m^{\text{s}}\right) \tag{11}$$

where $\text{KL}(.) = \sum_{i=1}^{N_1} \alpha_{m,i}^{\text{s}} \log(\alpha_{m,i}^{\text{s}}/\tilde{\alpha}_{m,i}^{\text{s}})$. In addition, we can integrate $\boldsymbol{\theta}_{\text{dec}}$ in the training process. With the teacher-forced cross entropy loss $\mathcal{L}_{\texttt{xent}}$, we define the **integrated training** loss:

$$\mathcal{L}_{\text{I}} = \mathcal{L}_{\texttt{xent}} + \lambda \mathcal{L}_{\text{KL}} \tag{12}$$

We cannot optimize $\mathcal{L}_{\text{I}}$ in an end-to-end fashion because the top-r operation in Eq. 6 is not differentiable. Hence, we interleave the training, i.e. update $\boldsymbol{\theta}_{\text{dec}}$ at fixed $\tilde{\boldsymbol{\theta}}$ and update $\tilde{\boldsymbol{\theta}}$ using $\mathcal{L}_{\text{KL}}$ only:

$$\Delta\boldsymbol{\theta}_{\text{dec}} = \nabla_{\boldsymbol{\theta}_{\text{dec}}}\mathcal{L}_{\text{I}}|_{\tilde{\boldsymbol{\theta}}} = \nabla_{\boldsymbol{\theta}_{\text{dec}}}\mathcal{L}_{\texttt{xent}}|_{\tilde{\boldsymbol{\theta}}} + \lambda\nabla_{\boldsymbol{\theta}_{\text{dec}}}\mathcal{L}_{\text{KL}}|_{\tilde{\boldsymbol{\theta}}} \tag{13}$$

$$\Delta\tilde{\boldsymbol{\theta}} = \nabla_{\tilde{\boldsymbol{\theta}}}\mathcal{L}_{\text{I}} = \cancelto{0}{\nabla_{\tilde{\boldsymbol{\theta}}}\mathcal{L}_{\texttt{xent}}} + \lambda\nabla_{\tilde{\boldsymbol{\theta}}}\mathcal{L}_{\text{KL}} \tag{14}$$

Because during training, we compute both $\alpha_{m,i}^{\text{s}}$ (ideal) and $\tilde{\alpha}_{m,i}^{\text{s}}$ (approx), we can use either in the top-r selection. Also, inspired by scheduled sampling (Bengio et al., 2015), we try mixing them: $\alpha_{m,i}^{\text{s}}$ with probability $1 - \frac{\text{step}}{\text{epoch\_size}}$, otherwise $\tilde{\alpha}_{m,i}^{\text{s}}$.

4.4 System Performance

In Table 3, we provide two vanilla baselines: *random* ($\mathcal{I}_m^{r,\text{Rnd}}$) obtained by random selection; *ideal* ($\mathcal{I}_m^{r,\text{Idl}}$) obtained by Eq. 6. The results show that the KL-only system clearly outperforms the random selection baseline, and the performance degradation of the KL-only system can be reduced by our integrated training. The results verify the *effectiveness* of our modified decoder that attends to a subset of sentences. Also, Table 3 shows that it is best to use $\mathcal{I}_m^{r,\text{Apx}}$ as reference in integrated training. This result is likely because we initialized integrated training from the KL-only model. In addition, we apply the modified architecture to BART trained on XSum (≤1k words), and to LoBART trained on Podcast and arXiv (≤4k words). The results in Fig. 2 confirm that the performance of our proposed method converges to that of the full attention baseline across all models and datasets.

System	Train	Inference	R1	R2	RL
Vanilla	✗	$\mathcal{I}_m^{r,\text{Rnd}}$	39.06	14.32	36.07
Vanilla	✗	$\mathcal{I}_m^{r,\text{Idl}}$	43.94	20.82	40.81
KL-only	$\mathcal{L}_{\text{KL}}$	$\mathcal{I}_m^{r,\text{Apx}}$	43.02	20.02	39.89
Int-Idl	$\mathcal{L}_{\text{I}}(\mathcal{I}_m^{r,\text{Idl}})$	$\mathcal{I}_m^{r,\text{Apx}}$	43.03	20.04	40.05
Int-Apx	$\mathcal{L}_{\text{I}}(\mathcal{I}_m^{r,\text{Apx}})$	$\mathcal{I}_m^{r,\text{Apx}}$	43.72	20.40	40.70
Int-Mix	$\mathcal{L}_{\text{I}}(\text{Mix})$	$\mathcal{I}_m^{r,\text{Apx}}$	43.31	20.21	40.35

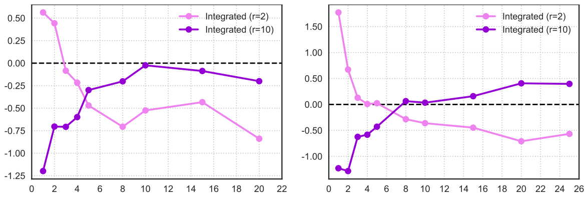
Table 3: Performance on CNNDM where r=5 for both training and inference. KL-only = $\tilde{\boldsymbol{\theta}}$ trained on $\mathcal{L}_{\text{KL}}$; Int = ($\boldsymbol{\theta}_{\text{dec}}$&$\tilde{\boldsymbol{\theta}}$) trained on $\mathcal{L}_{\text{I}}$. Rnd=random, Idl=ideal, Apx=approximation, Mix=scheduled(Idl/Apx).

In addition, $r^* \approx$ 5,10,30,30, respectively.[4] Although XSum has fewer sentences in average compared to CNNDM, $r^*_{\text{XSum}} > r^*_{\text{CNNDM}}$ as XSum is more abstractive. For longer summarization tasks as shown by Podcast and arXiv, the performance degradation appears larger, meaning that the task of constraining to salient sentences in longer tasks is more challenging, and larger r is required.

[4] r^* denotes r at which the ideal selection system's performance plateaus/reaches the full attn baseline's performance.

Sensitivity of r in Integrated Training

We train BART in three settings: r^{train}=2,5,10, and we show the performance level w.r.t. the model with r^{train}=5 in Fig. 4. The results show that setting r^{train} beyond r^* is not necessarily beneficial as shown by the model with r^{train}=10 in the CNNDM result, and it is best to set r^{train} close to $r^{\text{inference}}$.



(a) CNNDM

(b) XSum

Figure 4: ΔR1 (Y-axis) against r at inference (X-axis).

4.5 Further Discussion on Sentence-Level Encoder-Decoder Attention

The results in Section 4.4 demonstrate empirically that a neural network can predict sparsity, therefore, allowing sentence selection. Our novel framework requires an addition of modules to the original attention mechanism, and the real gain in speed will depend on the balance of the sparsity against the computational cost of the additional modules. Consequently, the challenge is to make these additional modules highly efficient. Because the particular network realization selected in this paper is to show the feasibility of our framework, its limitations and possible improvements are discussed as follows.

Limitations and Possible Improvements

The model choice of using RNN for the sentence encoder in Eq. 10 leads to a large additional computational cost (specifically large k_e) because the computational cost of RNN grows with $N_1 N_2 D^2$. Because the goal is to obtain a sentence-level representation, there is an opportunity to replace RNN by a hierarchical attention that runs over sentences, which could instead lead to a computational cost that grows with $N_1 N_2 D$. Additional sentence-level query and key mappings in Eq. 8 and Eq. 9 also incur a large computational cost.

Model-free Approximator

Lastly, we re-visit the sentence-level attention in Eq. 4, which have been approximated by Eq. 7 and the model-based approximator. It is a challenge to attempt via a model-free algebraic approximation, which does not require any training and has little additional inference-time cost. We examined various forms, and we present one model-free approach as well as experimental results in Appendix D, but the current form has worse summarization performance than our model-based approach.

5 Related Work

The discrepancy between low/moderate attention weight sparsity and good sparse approximation could be because a considerable amount of the attention weight is assigned to special tokens, e.g. ‘**.**’ in all sentences, but their vector norm is small, which was observed in Kobayashi et al. (2020).

The sparse attention (Eq. 3) with ideal selection (Eq. 6) can be considered as *content selection*, which has been shown to improve summarization (Gehrmann et al., 2018; Hsu et al., 2018). Recently, head-wise masks are applied to encoder-decoder attention at inference time, and a performance improvement is reported (Cao and Wang, 2021).

Voita et al. (2019) observed that heads are redundant, and Clark et al. (2019) found that a head in BERT rarely attends to several consecutive tokens. Based on these, Huang et al. (2021) applies a stride pattern in the encoder-decoder attention, reducing its cost by a factor of the stride size, and this method is likely complementary to our work.

6 Conclusion

We show that the computational cost of the transformer decoder becomes more significant at inference time. Towards reducing this cost, first, we show that there is sparsity in the encoder-decoder attention that allows us to reduce the computational cost with a minimal degradation. Second, we partition the sentence-level attention score, and we augment the standard decoder by adding a neural network to approximate the attention over sentences, allowing sentence selection. We show that the summarization performance of our approach converges to that of the full attention baseline, while switching the complexity from $\mathcal{O}(MN_1N_2)$ to $\mathcal{O}(MN_1 + k_w MrN_2 + k_e N_1 N_2)$.

Acknowledgments

This paper reports on research supported by ALTA institute, Cambridge Assessment English, University of Cambridge, and Cambridge International & St John’s College Scholarship. Thanks to Yiting Lu for interesting discussions. Thanks to the anonymous reviewers for their helpful comments.

References

Iz Beltagy, Matthew E Peters, and Arman Cohan. 2020. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*.

Samy Bengio, Oriol Vinyals, Navdeep Jaitly, and Noam Shazeer. 2015. Scheduled sampling for sequence prediction with recurrent neural networks. In *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pages 1171–1179.

Shuyang Cao and Lu Wang. 2021. Attention head masking for inference time content selection in abstractive summarization. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5008–5016, Online. Association for Computational Linguistics.

Kyunghyun Cho, Bart van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using RNN encoder–decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734, Doha, Qatar. Association for Computational Linguistics.

Kevin Clark, Urvashi Khandelwal, Omer Levy, and Christopher D. Manning. 2019. What does BERT look at? an analysis of BERT's attention. In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 276–286, Florence, Italy. Association for Computational Linguistics.

Ann Clifton, Sravana Reddy, Yongze Yu, Aasish Pappu, Rezvaneh Rezapour, Hamed Bonab, Maria Eskevich, Gareth Jones, Jussi Karlgren, Ben Carterette, and Rosie Jones. 2020. 100,000 podcasts: A spoken English document corpus. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 5903–5917, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Arman Cohan, Franck Dernoncourt, Doo Soon Kim, Trung Bui, Seokhwan Kim, Walter Chang, and Nazli Goharian. 2018. A discourse-aware attention model for abstractive summarization of long documents. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 615–621, New Orleans, Louisiana. 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 *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)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

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, Brussels, Belgium. Association for Computational Linguistics.

Karl Moritz Hermann, Tomás Kociský, 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 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pages 1693–1701.

Wan-Ting Hsu, Chieh-Kai Lin, Ming-Ying Lee, Kerui Min, Jing Tang, and Min Sun. 2018. A unified model for extractive and abstractive summarization using inconsistency loss. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 132–141, Melbourne, Australia. Association for Computational Linguistics.

Luyang Huang, Shuyang Cao, Nikolaus Parulian, Heng Ji, and Lu Wang. 2021. Efficient attentions for long document summarization. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1419–1436, Online. Association for Computational Linguistics.

Rosie Jones, Ben Carterette, Ann Clifton, Maria Eskevich, Gareth J. F. Jones, Jussi Karlgren, Aasish Pappu, Sravana Reddy, and Yongze Yu. 2020. Trec 2020 podcasts track overview. In *The 29th Text Retrieval Conference (TREC) notebook*.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Goro Kobayashi, Tatsuki Kuribayashi, Sho Yokoi, and Kentaro Inui. 2020. Attention is not only a weight: Analyzing transformers with vector norms. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7057–7075, Online. Association for Computational Linguistics.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational*

Linguistics, pages 7871–7880, Online. Association for Computational Linguistics.

Manling Li, Lingyu Zhang, Heng Ji, and Richard J. Radke. 2019. Keep meeting summaries on topic: Abstractive multi-modal meeting summarization. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2190–2196, Florence, Italy. Association for Computational Linguistics.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Potsawee Manakul and Mark Gales. 2021. Long-span summarization via local attention and content selection. 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)*, pages 6026–6041, Online. Association for Computational Linguistics.

Potsawee Manakul, Mark J.F. Gales, and Linlin Wang. 2020. Abstractive Spoken Document Summarization Using Hierarchical Model with Multi-Stage Attention Diversity Optimization. In *Proc. Interspeech 2020*, pages 4248–4252.

Shashi Narayan, Shay B. Cohen, and Mirella Lapata. 2018. Don't give me the details, just the summary! topic-aware convolutional neural networks for extreme summarization. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1797–1807, Brussels, Belgium. Association for Computational Linguistics.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 8024–8035.

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.

Yi Tay, Mostafa Dehghani, Dara Bahri, and Donald Metzler. 2020. Efficient transformers: A survey. *arXiv preprint arXiv:2009.06732*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008.

Elena Voita, David Talbot, Fedor Moiseev, Rico Sennrich, and Ivan Titov. 2019. Analyzing multi-head self-attention: Specialized heads do the heavy lifting, the rest can be pruned. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5797–5808, Florence, Italy. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*.

Manzil Zaheer, Guru Guruganesh, Kumar Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, and Amr Ahmed. 2020. Big bird: Transformers for longer sequences. In *Advances in Neural Information Processing Systems*, volume 33, pages 17283–17297. Curran Associates, Inc.

A Reproducibility Details

A.1 Models

BART/LoBART: We use the HuggingFace's implementation (Wolf et al., 2020), including BART models fine-tuned to CNNDM[5] and XSum[6]. We take LoBART from Manakul and Gales (2021), including LoBART(4k)+MCS fine-tuned to Podcast and arXiv. MCS is the multitask content selection system for handling the Podcast/arXiv input documents that exceed 4096 words.

Modified Architecture: As shown in Fig. 3, the modified architecture consists of a sentence encoder and a sentence-level encoder-decoder attention. The sentence encoder, which approximates $f_2(.)$ in Eq. 7, is a two-layer bi-directional GRU (Cho et al., 2014) with hidden dimension of 1024. The sentence-level encoder-decoder attention has linear mapping weights $\mathbf{W}^{\mathtt{Q}}$ (query) and $\mathbf{W}^{\mathtt{K}}$ (key), and both weights have the same dimension as and are initialized from their corresponding word-level encoder-decoder attention's linear mapping weights in the base model. The total number of additional parameters is 58.8M.

A.2 Data

CNNDM: We follow the standard train/valid/test split of 287,113/13,368/11,490.

XSum: We follow the standard train/valid/test split of 204,045/11,332/11,334.

Spotify Podcast: We follow the data processing and split in Jones et al. (2020), resulting in train/valid/test splits of 60,415/2,189/1,027.

arXiv: We follow the standard train/valid/test split of 203,037/6,436/6,440.

Our data processing is based on the byte-pair-encoding (BPE) tokenizer same as the BART-large tokenizer, and we use the NLTK toolkit for sentence splitting. **In Fig. 2 and Fig. 4, to reduce computational cost, we use first 2,000 samples of each test set, except Podcast which contains less than 2,000 samples.

A.3 Training and Inference

We use PyTorch (Paszke et al., 2019) in our experiments. All training experiments use the Adam optimizer (Kingma and Ba, 2015) with β_1=0.9, β_2=0.999, and the learning rate is:

$$\text{lr} = 0.002 \times \min(\text{step}^{-0.5}, \text{step} \times \text{warmup}^{-1.5})$$

where we use 20,000 warmup steps. In all experiments, we set batch size to 1, and gradient accumulation to 2 steps. We evaluate the training loss on the validation set every 20,000 steps, and stop the training if the validation loss does not improve 3 times. All training experiments converged within 1 epoch. All experiments were carried out in 32-bit precision on either one V100 (32GB) GPU, or one RTX 2080Ti (11GB) GPU.

Dataset	N	N_1	M	N/M
CNNDM	870	28.7	67.4	14.2
XSum	489	17.4	27.9	18.2
Podcast	5727	330.8	86.6	143.9
arXiv	8584	237.0	364.9	45.3

Table 4: Data statistics (average over corpus). N=input length, N_1=#sentences, M=summary length.

At inference time, we use the standard setting: beam search of width 4, and length penalty of 2.0 (Wu et al., 2016) for all experiments. The ROUGE (Lin, 2004) scoring tool is `pyrouge`.[7]

A.4 Multi-head attention

In all of the equations and expressions in the paper, we omit the heads for simplicity. Both BART and LoBART models have 16 heads. In Fig. 1, we average $\alpha^{\mathsf{s}}_{m,i}$ over heads, before the summation. When computing an uncertainty measure such as entropy H(.) or KL-divergence KL(.), we compute the measure for each head separately and take the average. In obtaining $\mathcal{I}^r_m$, we average $\alpha^{\mathsf{s}}_{m,i}$ over heads, before the top-r operation, i.e. all heads get assigned the same subset of sentences, but the differences are across layers and decoding timesteps.

A.5 KL Loss and Integrated Training

The target $\alpha^{\mathsf{s}}_{m,i}$ is re-normalized to encourage higher sparsity as follows:

$$\boldsymbol{\alpha}^{\mathsf{s}}_m \leftarrow \text{softmax}\left(\frac{\log(\boldsymbol{\alpha}^{\mathsf{s}}_m)}{T}\right) \qquad (15)$$

where temperature T is set to 0.5. For integrated training, we set λ in $\mathcal{L}_{\mathtt{I}}$ to 0.2. We initialize integrated training experiments from KL-only models.

[5] https://huggingface.co/facebook/bart-large-cnn

[6] https://huggingface.co/facebook/bart-large-xsum

[7] https://pypi.org/project/pyrouge/

B Time Analysis

For each mode in Table 1, we take 6 samples of M and the average time of 100 iterations. Curve fitting yields R-squared of at least 0.994.

Computational time as function of M and N is $\texttt{time} = c_1 + c_2 M + c_3 N + c_4 MN + c_5 M^2 + c_6 N^2$. The coefficients are obtained by a least-squares regression, e.g. $\mathbf{c}^* = (\mathbf{P}^T \mathbf{P})^{-1} \mathbf{P}^T \mathbf{t}$ where $\mathbf{P}$ is the matrix of M, N associated with the coefficients, and $\mathbf{t}$ contains the time measures. We collect 30 samples of the average time of 100 F+B passes, spanning $N \in [256, 1024]$ and $M \in [50, 300]$. The normalized coefficients are: $c_1 = 1.00, c_2 = 3.78 \times 10^{-3}, c_3 = 3.15 \times 10^{-3}, c_4 = 1.47 \times 10^{-6}, c_5 = 7.26 \times 10^{-7}, c_6 = 7.79 \times 10^{-7}$. Because $c_4 \approx 2c_5$ and $N > M$, the enc-dec attention cost is greater than the decoder self attention cost.

C Sensitivity to Random Selection

System	R1	R2	RL
Vanilla ($\gamma = 0.0$)	39.06	14.32	36.07
$\mathcal{L}_{\mathtt{A}}$-tuned ($\gamma = 0.1$)	28.43	7.72	24.00
$\mathcal{L}_{\mathtt{A}}$-tuned ($\gamma = 1.0$)	21.38	4.22	17.69

Table 5: Impact of sparsity on the sensitivity to random selection based on CNNDM with $r = 5$.

D Model-free Approximation for Eq. 4

Since $\alpha^{\mathtt{s}}_{m,i} = \frac{1}{Z_m} \sum_{j=1}^{J_i} \exp(\mathbf{q}_m \cdot \mathbf{k}_{i,j})$ is used to rank input sentences, the normalization term, Z_m, can be dropped. The encoder-decoder attention has $\mathcal{O}(MN_1N_2)$ complexity because $\mathbf{q}_m \cdot \mathbf{k}_{i,j}$ is computed for every m and (i, j) pair. Hence, if we can group $\mathbf{k}_{i,j}$ into sentences, the complexity could potentially be reduced. We try the following approximation of unnormalized $\alpha^{\mathtt{s}}_{m,i}$:

$$\sum_{j=1}^{J_i} \exp(\mathbf{q}_m \cdot \mathbf{k}_{i,j})$$

$$= \sum_{j=1}^{J_i} \prod_{d=1}^{D} \exp(q_{m,d} k_{i,j,d}) \tag{16}$$

$$\approx \sum_{j=1}^{J_i} \sum_{d=1}^{D} \exp(q_{m,d} k_{i,j,d}) \tag{17}$$

$$\approx \sum_{j=1}^{J_i} \sum_{d=1}^{D} \phi(q_{m,d}) \phi(k_{i,j,d}) \tag{18}$$

$$= \phi(\mathbf{q}_m) \cdot \left(\sum_{j=1}^{J_i} \phi(\mathbf{k}_{i,j}) \right) \tag{19}$$

where $d = \{1, ..., D\}$ is the hidden dimension, and $\phi(.) = \text{ELU}(.) + 1$ (or other form such as exp and ReLU). The model-free method reduces complexity from $\mathcal{O}(MN_1N_2)$ to $\mathcal{O}(MN_1 + k_w MrN_2 + k_e N_1 N_2)$ where k_e is now much smaller compared to the model-based approach. Based on Eq. 19, we provide model-free results in Table 6.

Selection Method	R1	R2	RL
Ideal (Eq.6)	43.94	20.82	40.81
Best Model-based	43.72	20.40	40.70
Random	39.06	14.32	36.07
Model-free (Eq.19)	40.01	17.28	36.95

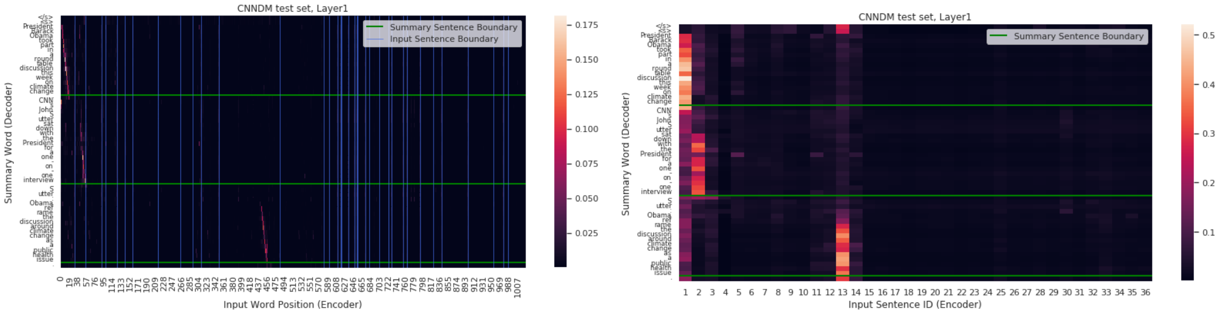
Table 6: Model-free results on CNNDM ($r = 5$).

Our model-free approach is better than the random selection baseline, but it is significantly worse than both the ideal selection baseline and the model-based approach. The reasons for this poor performance are: (i) the approximation from (16) to (17) requires the following condition to be true: $A_1 A_2 > B_1 B_2 \rightarrow A_1 + A_2 > B_1 + B_2$, so it is inaccurate when the values are not in a similar range; (ii) the approximation from (17) to (18) is inaccurate for non-positive values. In conclusion, this experiment investigates an alternative challenging method, which would not require any training and would be computationally cheaper at inference time. Although the current algebra does not work well, we hope that our initial study might draw more interests into this type of model-free approach to exploit the sentence structure in seq2seq tasks such as abstractive summarization.

E Word-level and Sentence-level Attention Weight Plots

Constraining the encoder-decoder attention to r sentences is motivated by the observations of sentence-level attention (in Fig. 5b, 6b, 7b, 8b). Note that we average over all heads for the plots.

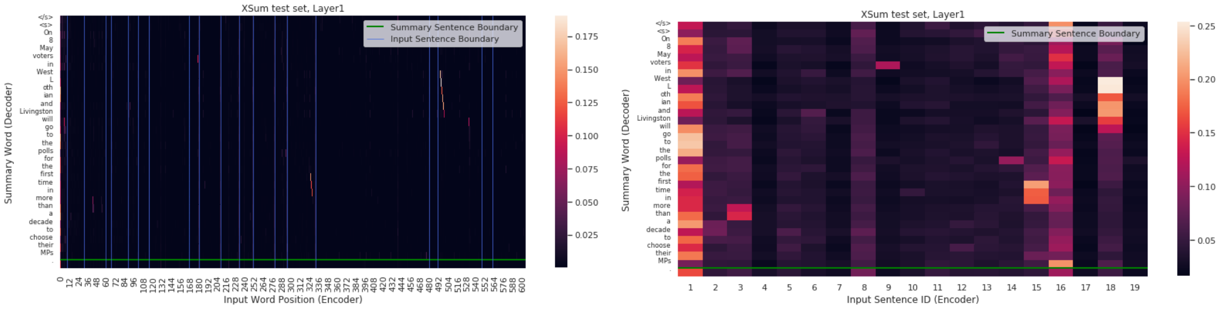
For instance, Fig. 5 shows that the decoder attends particularly to input sentences #1,#2,#13 in the summary generation. Compared to Fig. 5, Fig. 6 shows a wider spread of the attention over sentences in a more abstractive task. When using LoBART, Fig. 7 and Fig. 8 show a similar trend of the sparsity to BART scenarios. These figures also explain Fig. 1a (Section 3) that $\sum_i^{\mathcal{I}^r_m} \alpha^{\mathtt{s}}_{m,i}$ is only low/moderate because most sentences get assigned some attention weights, despite being non-salient.



(a) Word-Level

(b) Sentence-Level ($\alpha^{\text{s}}_{m,i}$)

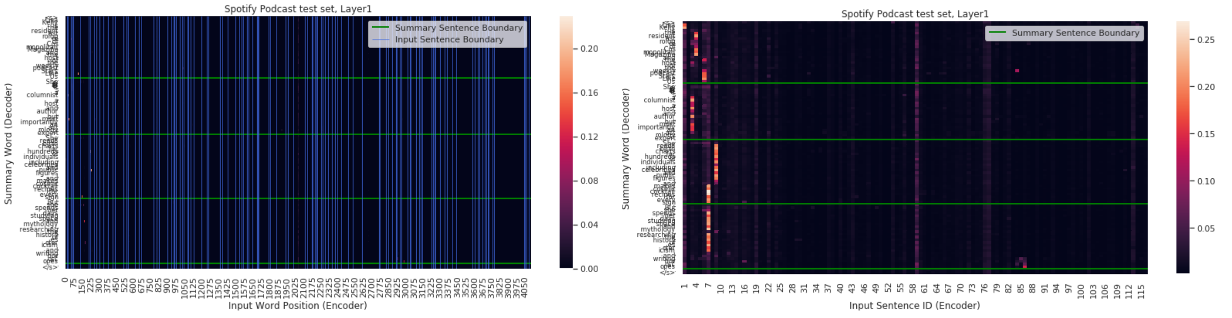
Figure 5: Example of BART's encoder-decoder attention evaluated on CNNDM test set.



(a) Word-Level

(b) Sentence-Level ($\alpha^{\text{s}}_{m,i}$)

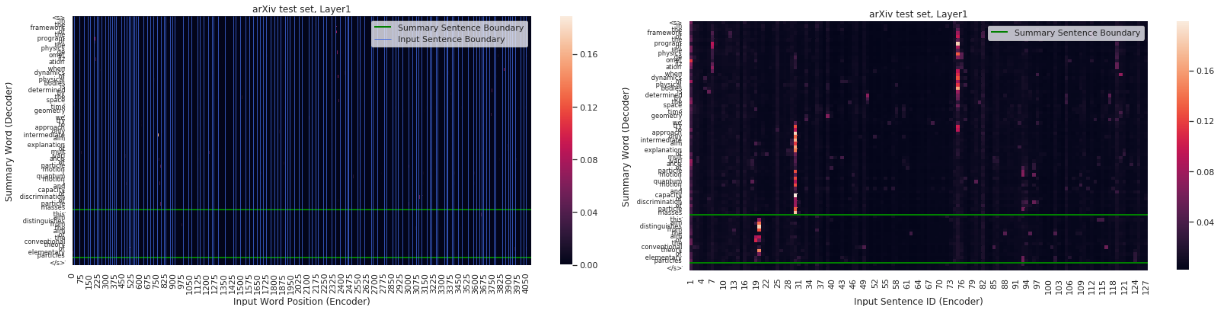
Figure 6: Example of BART's encoder-decoder attention evaluated on XSum test set.



(a) Word-Level

(b) Sentence-Level ($\alpha^{\text{s}}_{m,i}$)

Figure 7: Example of LoBART's encoder-decoder attention evaluated on Podcast test set.



(a) Word-Level

(b) Sentence-Level ($\alpha^{\text{s}}_{m,i}$)

Figure 8: Example of LoBART's encoder-decoder attention evaluated on arXiv test set.