DQ-BART: Efficient Sequence-to-Sequence Model via Joint Distillation and Quantization

Zheng Li[†§1] **Zijian Wang**[§2] **Ming Tan**[2] **Ramesh Nallapati**[2] **Parminder Bhatia**[2]
Andrew Arnold[2] **Bing Xiang**[2] **Dan Roth**[2,3]
[1]Cornell University [2]AWS AI Labs [3] University of Pennsylvania
zl634@cornell.edu {zijwan, mingtan}@amazon.com
{rnallapa, parmib, anarnld, bxiang, drot}@amazon.com

Abstract

Large-scale pre-trained sequence-to-sequence models like BART and T5 achieve state-of-the-art performance on many generative NLP tasks. However, such models pose a great challenge in resource-constrained scenarios owing to their large memory requirements and high latency. To alleviate this issue, we propose to jointly distill and quantize the model, where knowledge is transferred from the full-precision teacher model to the quantized and distilled low-precision student model. Empirical analyses show that, despite the challenging nature of generative tasks, we were able to achieve a 16.5x model footprint compression ratio with little performance drop relative to the full-precision counterparts on multiple summarization and QA datasets. We further pushed the limit of compression ratio to 27.7x and presented the performance-efficiency trade-off for generative tasks using pre-trained models. To the best of our knowledge, this is the first work aiming to effectively distill and quantize sequence-to-sequence pre-trained models for language generation tasks.

1 Introduction

Pretrained sequence-to-sequence (seq2seq) models such as BART (Lewis et al., 2020; Liu et al., 2020) and T5 (Raffel et al., 2020; Xue et al., 2021) have shown great success in various natural language processing (NLP) tasks, such as text summarization (Nallapati et al., 2016; See et al., 2017; Narayan et al., 2018), machine translation, question answering (Fan et al., 2019) and information extraction (Zhou et al., 2021). However, such large-scale pre-trained language models come with hundreds of millions of parameters: Lewis et al. (2020) trained a BART model with 400M parameters, while Raffel et al. (2020) pushed the limit to 11 billion parameters in T5.

The continual growth in model sizes leads to significant demand in both computation and memory resources during inference, and poses a huge challenge on deployment, especially in real-time and/or resource-constrained scenarios. This motivates researchers to compress large pre-trained models to be smaller and faster while retaining strong performance. Among existing compression approaches such as weight-sharing (Dehghani et al., 2019; Lan et al., 2020), low-rank approximation (Ma et al., 2019; Lan et al., 2020), and pruning (Michel et al., 2019), quantization approaches have received attention recently since they reduce model footprints using lower bits for the weight values without changing the carefully-designed model architecture. Most prior work on transformer quantization focused on BERT-based transformers (Zhang et al., 2020; Zafrir et al., 2019; Bai et al., 2021). However, efficient quantization on the encoder-decoder transformers is insufficiently studied. Prato et al. (2020) achieve 8-bit quantization for a seq2seq transformer without significant loss of performance but low-bit quantization proved to be difficult for this model (4-bit performance in Table 2 in their work) due to the accumulation of quantization errors in seq2seq models. Moreover, their work did not target quantizing large-scale pre-trained language models, nor could it be applied to other NLP tasks besides machine translation. Meanwhile, model distillation which transfers knowledge from a large teacher model to a smaller student model has been widely investigated for BERT compression (Sanh et al., 2019; Jiao et al., 2020).

Recently, Shleifer and Rush (2020) applied "shrink and fine-tune" distillation method on BART for text summarization, yet their work focuses more on the methodology for distilling text summarization only. Besides, their work did not yield a sig-

[†]Work done during an internship at AWS AI Labs.
[§]Equal contribution.

Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics
Volume 2: Short Papers, pages 203 - 211

nificant model footprint reduction, one of the most challenging issues in the deployment of large models in resource-constrained scenarios.

In this work, we try to address the challenge of building a more efficient seq2seq model by answering two research questions: first, how well does the quantized seq2seq model perform on various tasks? Second, how do we combine quantization and distillation to push the limit of compressing the seq2seq model without significant performance losses in challenging tasks like summarization and question answering? To this end, we proposed a joint distillation and quantization framework, which efficiently transfers the knowledge from a full-precision teacher seq2seq model to its student with fewer layers and ultra-low bits for encoding its parameters. Experimental results on BART show that the proposed models reduce the model footprint by 16.5x while preserving competitive performances on multiple language generation benchmarks, and further illustrate the performance-efficiency trade-off of compressing seq2seq models up to 27.7x smaller. To the best of our knowledge, this is the first work aiming to effectively distill and quantize seq2seq pre-trained models for language generation tasks.

2 Distilling and Quantizing BART

In this section, we consider two directions for reducing the size of our generative language model: quantization (§2.1) and distillation (§2.2). We apply distillation-aware training (§2.3) to train a quantized and distilled low-precision model as a student model to emulate the full-precision teacher model.

2.1 Quantization

Quantization refers to the operation of mapping a real (high-precision) number to its low-precision counterpart in order to achieve model footprint reduction. There has been extensive study on applying quantization to training neural networks. Different quantization schemes include, e.g., linear quantization (e.g., Hubara et al., 2016, 2017; Jacob et al., 2018), non-linear quantization (Li and Sa, 2019), approximation-based quantization method (Lin et al., 2016), and loss-aware quantization (Hou and Kwok, 2018). In our work, we used the approximation-based method with linear quantization following Zhang et al. (2020).

Quantizing BART We applied quantization to the weights of all the hidden layers and most of the embeddings. Following previous work (Zhang et al., 2020), we did not quantize positional embeddings and quantized activations only to 8 bits.

Weight Quantization We dive into the mathematical details of how to quantize the weights in BART models. Let us denote $\mathbf{w}^t \in \mathcal{R}^{n_t}$ as the vector obtained by stacking all the columns of the full-precision weight matrix $\mathbf{W}^t$ that we wish to quantize at iteration t. By quantizing $\mathbf{w}^t$, we are looking for a scaling factor (also known as quantization step) α^t and a low-precision number $\mathbf{b}^t$, to replace full precision weight $\mathbf{w}^t$ with $\alpha^t\mathbf{b}^t$. When quantizing with more than 2 bits, we are applying the commonly used symmetric linear quantization, with

$$\begin{aligned}\alpha^t &= \max_i |w_i^t| \,/\, th \\ \mathbf{b}^t &\in \{-th, \cdots, -1, 0, 1, \cdots, th\}^{n_t}\end{aligned}$$

where $th = 2^{n_b-1} - 1$ and n_b is the number of bits we use for quantization. Then $\mathbf{b}^t$ can be obtained by $\mathbf{b}^t = round(\mathbf{w}^t/\alpha^t)$. When quantizing with 2 bits, we use the approximation based TWN method (Li et al., 2016). The mathematical details are provided in Appendix A.

2.2 Distillation

The second task we consider is knowledge distillation, where we train a smaller student model to mimic the behavior of a larger teacher model; specifically, we want to reproduce the output logits, attentions, and hidden states of the teacher model. Following Shleifer and Rush (2020), we initialize the student model by copying the weights from maximally spaced layers of the teacher model, e.g., when initializing a 3-layer student encoder (decoder) from a 6-layer teacher encoder (decoder), we copy the 0th, 3th and 5th layers from the teacher to the student. When copying only 1 layer, we choose the last instead of the first, which has been shown empirically to yield better performance. Different than Shleifer and Rush (2020) who only distill the decoder, we distill both the encoder and the decoder. After initialization, we fine-tune the student model with the combined objective of task loss and distillation loss, i.e. $\mathcal{L}_{\text{data}} + \mathcal{L}_{\text{dist}}$, with

$$\mathcal{L}_{\text{dist}} = \mathcal{L}_{\text{logits}} + \mathcal{L}_{\text{att}} + \mathcal{L}_{\text{hid}}$$

where the RHS are MSE losses measuring the difference between the student and teacher with regard to output logits, attention scores (including

Model	Size (MB)	Summarization						Long-form QA		
		CNN/DailyMail			XSUM			ELI5		
W-E-A (#bits) E-D (#layers)		R1	R2	RL	R1	R2	RL	R1	R2	RL
32-32-32 6-6	531 (1x)	44.90	22.25	42.09	43.84	20.79	35.71	26.02	5.11	15.36
8-8-8 6-6 (direct quant.)	137 (3.9x)	11.36	1.01	11.01	22.74	5.69	17.81	6.72	0.43	4.89
Distillation-Aware Quantization										
8-8-8 6-6	137 (3.9x)	44.66	21.92	41.86	42.51	19.61	34.61	27.10	5.15	16.23
2-2-8 6-6	39 (13.6x)	42.94	20.07	40.13	40.06	17.34	32.46	26.33	4.97	16.15
Distillation-Aware Quantization + Distillation										
8-8-8 6-3	110 (4.8x)	43.99	21.25	41.24	41.94	19.21	34.21	26.38	5.13	16.27
8-8-8 6-1	92 (5.8x)	42.52	20.04	40.05	39.42	17.70	32.69	24.27	4.74	15.71
8-8-8 3-1	72 (7.4x)	41.18	18.75	38.58	36.39	15.29	29.91	23.69	453	15.51
2-2-8 6-3	32 (16.5x)	42.49	19.71	39.70	39.66	17.26	32.33	25.41	4.83	15.94
2-2-8 6-1	27 (19.2x)	41.14	18.72	38.66	36.61	15.33	30.22	23.34	4.31	15.20
2-2-8 3-1	22 (23.5x)	40.14	17.75	37.60	33.56	13.05	27.48	22.60	3.99	14.95
2-2-8 1-1	19 (27.7x)	39.00	16.73	36.42	29.04	9.56	23.47	21.51	3.44	14.30

Table 1: Distillation and quantization results on BART for text summarization on CNN/DailyMail and XSUM benchmarks and long-form question answering on the ELI5 benchmark. We abbreviate the number of bits for **w**eights, word **e**mbedding and **a**ctivations as "W-E-A (#bits)", followed by the number of **e**ncoder and **d**ecoder layers as "E-D (#layers)". We use the rouge-{1,2,L} as evaluation metrics (Lin, 2004). We found that distillation-aware quantized models achieves comparable or even better performance compared with the full precision models, and combining quantization and distillation, e.g., from "2-2-8 6-6" to "2-2-8 6-3", gives us a further boost in model footprint compression ratio without significant sacrifice in performance. See §3.2 for details.

encoder attention, decoder attention and cross attention), and hidden states (including all encoder and decoder layers).[1] We include the details of the loss in Appendix B for completeness.

2.3 Distillation-aware quantization

To fine-tune our quantized and distilled model, we use the technique of distillation-aware quantization with a teacher-student architecture from (Zhang et al., 2020)[2]. We treat the quantized and distilled low-precision model as a student model trained to emulate the full precision model, which in this case is the teacher model. Meanwhile, we also keep the full-precision distilled counterpart of the student model for parameter update. At each iteration, we first quantize the full precision student model to get its quantized version, then do the forward pass with the low-precision student model and get the task loss as well as the distillation losses discussed in §2.2. Finally, we use these losses to update the parameters in the full-precision student model.

[1]Based on an initial small-scale study, we didn't find a significant difference between weighted and unweighted losses in our setting. For simplicity, we use unweighted loss here and leave the tuning of weights for future work.

[2]Note that in this work we jointly distill and quantize encoder-decoder models, while Zhang et al. (2020) used a similar technique but 1) for quantizing encoder-only models and 2) without the actual model distillation.

3 Experiments and Discussions

In this section, we evaluate the efficacy of jointly Distilling and Quantizing BART (hereinafter, DQ-BART) on text summarization and long-form question answering using three benchmarks: CNN/DailyMail (See et al., 2017), XSUM (Narayan et al., 2018), and ELI5 (Fan et al., 2019). We additionally study machine translation with mBART on WMT English-Romanian (En-Ro) (Bojar et al., 2016).

3.1 Experimental Setup

We followed the standard splits of these datasets. The statistics could be found in Appendix C. For ELI5, we reproduced the author's implementation to train a dense retriever that retrieves 10 supporting documents from Wikipedia for each question. Additional details could be found in Appendix D.

As our target is achieving efficient seq2seq generative models, we used base-sized BART for summarization and question answering tasks. For machine translation, we used mBART-large due to the lack of pretrained base-sized multilingual BART models. We reused existing models[3], and finetuned our own models on end tasks when no open-sourced model is available. We trained our quantized-only models for 10 epochs and distilled-and-quantized

[3]https://huggingface.co/ainize/bart-base-cnn; https://huggingface.co/facebook/mbart-large-en-ro

models for 20 epochs. We used a batch size of 128, a learning rate of 3×10^{-5} with 5% linear warmup, and selected the best model based on rouge-L scores on the development set. We set generative hyperparameters following previous work (Lewis et al., 2020). All experiments were performed on A100 GPUs.

3.2 DQ-BART Results and Discussions

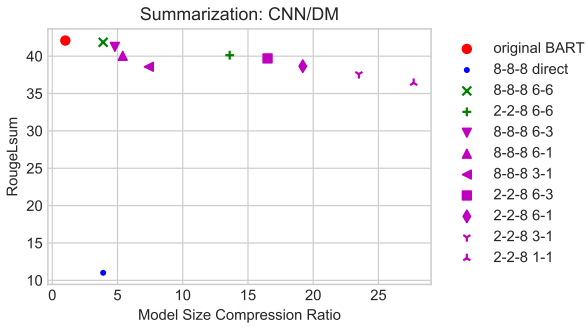

Figure 1: Visualization of performance v.s. model footprint compression ratio on CNN/DailyMail based on Table 1. Green dots are for quantization only, and purple dots are for distillation + quantization. We found that the performance degradation is minimal as the compression ratio grows, especially before 20x.

We summarized the main results in Table 1 and visualized the performance on text summarization on the CNN/DailyMail dataset in Figure 1. Additional visualizations are in Appendix E. We found that:

1. Direct quantization performs poorly in generation tasks. The rouge-L score drops ~50-75% relatively compared with the baseline.
2. The performance of 8-bit distillation-aware quantized models ("8-8-8 6-6") achieves comparable or even better performance compared with the full precision models across all tasks, signaling that 8-bit is not too challenging for generative models like BART, similar to the findings for BERT (Zhang et al., 2020).
3. We were able to achieve a 13.6x model size compression ratio when using 2-bit quantization with the trade-off of slight performance drop for summarization tasks and even no performance drop for the long-form QA task.
4. Combining quantization and distillation gives us a further boost in model compression ratio without significant further sacrifice in performance. For example, when using 2-bit quantization, by cutting the layers of the decoder in half (from "2-2-8 6-6" to "2-2-8 6-3"), we only saw < 0.5 rouge-L performance drop across all tasks while getting another 2.9x compression.
5. When pushing the compression rate to the limit ("2-2-8 1-1"), we were able to achieve a 27.7x compression ratio while still preserving reasonable performance. We observed a rouge-L drop of 5.67 for CNN/DailyMail (42.09 $\rightarrow$ 36.42), 12.24 for XSUM (35.71 $\rightarrow$ 23.47), and 1.06 for ELI5 (15.36 $\rightarrow$ 14.30). Thus, for certain tasks a large model compression ratio would not lead to a significant performance drop while for others the drop could be huge, suggesting that the specific compression ratio to use should be decided on a task-by-task basis with the trade-off of performance and efficiency in mind.

3.3 DQ-mBART for Translation

We further extend our study to see how distillation and quantization work for mBART (Liu et al., 2020), a deeper multilingual model. We experimented mBART-large on WMT English-Romanian translation task (Bojar et al., 2016). The results are in Table 2.

Model	Size	BLEU
32-32-32 12-12	1x	26.82
8-8-8 12-12 (direct quant.)	4.0x	0.01
Distillation-Aware Quantization		
8-8-8 12-12	4.0x	25.91
2-2-8 12-12	15.2x	23.48
Distillation-Aware Quantization + Distillation		
8-8-8 12-6	4.7x	25.61
8-8-8 12-3	5.2x	24.22
8-8-8 12-1	5.6x	20.61
2-2-8 12-6	18.0x	17.66
2-2-8 12-3	19.9x	16.99
2-2-8 12-1	21.3x	12.81
2-2-8 1-1	30.6x	10.36

Table 2: Distillation and quantization results for translation on WMT16 En-Ro with mBART-large.

We found that distillation-aware quantization yields reasonably good performance, similar to the findings in DQ-BART (Table 1). However, the performance drops substantially when performing 2-bit quantization with distillation, possibly due to the accumulation of the distillation/quantization error becoming more significant with deeper models and the challenging nature of machine translation.

Future work may explore how to improve the performance of joint distillation and quantization for deep models under a low-bit setting.

3.4 Distillation and Quantization v.s. Distillation Only

	Model	Size	R1	R2	RL
CNN/DM	Distill + Quant 8-8-8 6-3	**4.8x**	**43.99**	**21.25**	**41.24**
	Distill only 16-16-16 3-1	3.8x	41.17	18.72	38.62
XSUM	Distill + Quant 8-8-8 6-3	**4.8x**	**41.94**	**19.21**	**34.21**
	Distill only 16-16-16 3-1	3.8x	36.60	15.46	30.07
ELI5	Distill + Quant 8-8-8 6-3	**4.8x**	**26.38**	**5.13**	**16.27**
	Distill only 16-16-16 3-1	3.8x	23.80	4.54	15.40

Table 3: Comparisons between distillation-only and joint distillation and quantization.

We want to understand how much gain there is when doing joint distillation and quantization compared with distillation-only method (Shleifer and Rush, 2020). To do so, we trained distillation-only models and compared them with DQ-BART with a similar size. From Table 3, we found that joint distillation and quantization performs much better across all tasks, signaling the huge gain with joint distillation and quantization. Additional ablation study on "Shrink and Finetune" could be found in Appendix F.

4 Conclusion

Transformer-based pre-trained seq2seq language models like BART have greatly advanced the state of the art in a range of NLP tasks. Yet, these extremely large-scale models pose a challenge in resource-constrained scenarios. To alleviate this issue, we proposed DQ-BART, a jointly distilled and quantized BART model. Empirical results show that, despite the difficult nature of language generation tasks, we achieve a 16.5x model footprint compression ratio with little performance drop on three generative benchmarks, and further present the performance-efficiency trade-off for seq2seq models up to a 27.7x compression ratio. Additionally, we studied distillation and quantization for mBART on a machine translation task, and highlighted the challenge of joint low-bit quantization with distillation for deeper models on cross-lingual tasks. To the best of our knowledge, our method is the first to apply joint quantization and distillation on pretrained language models, and this is the first work aiming to effectively distill and quantize seq2seq pretrained models for language generation tasks. We hope this work could open doors for developing and applying efficient seq2seq language models. We leave additional compression methods like attention head pruning (Michel et al., 2019) and sequence-level distillation (Kim and Rush, 2016), and the measurement of latency improvements in various settings for future work. Our code is available at `https://www.github.com/amazon-research/dq-bart/`.

Acknowledgment

We would like to thank colleagues at AWS AI Labs and our anonymous ARR reviewers for their constructive feedback.

References

Haoli Bai, Wei Zhang, Lu Hou, Lifeng Shang, Jin Jin, Xin Jiang, Qun Liu, Michael Lyu, and Irwin King. 2021. BinaryBERT: Pushing the limit of BERT quantization. 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 4334–4348, Online. Association for Computational Linguistics.

Ondřej Bojar, Rajen Chatterjee, Christian Federmann, Yvette Graham, Barry Haddow, Matthias Huck, Antonio Jimeno Yepes, Philipp Koehn, Varvara Logacheva, Christof Monz, Matteo Negri, Aurélie Névéol, Mariana Neves, Martin Popel, Matt Post, Raphael Rubino, Carolina Scarton, Lucia Specia, Marco Turchi, Karin Verspoor, and Marcos Zampieri. 2016. Findings of the 2016 conference on machine translation. In *Proceedings of the First Conference on Machine Translation: Volume 2, Shared Task Papers*, pages 131–198, Berlin, Germany. Association for Computational Linguistics.

Mostafa Dehghani, Stephan Gouws, Oriol Vinyals, Jakob Uszkoreit, and Lukasz Kaiser. 2019. Universal transformers. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Angela Fan, Yacine Jernite, Ethan Perez, David Grangier, Jason Weston, and Michael Auli. 2019. ELI5: Long form question answering. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3558–3567, Florence, Italy. Association for Computational Linguistics.

Lu Hou and James T. Kwok. 2018. Loss-aware weight quantization of deep networks. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net.

Itay Hubara, Matthieu Courbariaux, Daniel Soudry, Ran El-Yaniv, and Yoshua Bengio. 2016. Binarized neural networks. In *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*, pages 4107–4115.

Itay Hubara, Matthieu Courbariaux, Daniel Soudry, Ran El-Yaniv, and Yoshua Bengio. 2017. Quantized neural networks: Training neural networks with low precision weights and activations. *The Journal of Machine Learning Research*, 18(1):6869–6898.

Benoit Jacob, Skirmantas Kligys, Bo Chen, Menglong Zhu, Matthew Tang, Andrew G. Howard, Hartwig Adam, and Dmitry Kalenichenko. 2018. Quantization and training of neural networks for efficient integer-arithmetic-only inference. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pages 2704–2713. IEEE Computer Society.

Xiaoqi Jiao, Yichun Yin, Lifeng Shang, Xin Jiang, Xiao Chen, Linlin Li, Fang Wang, and Qun Liu. 2020. TinyBERT: Distilling BERT for natural language understanding. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4163–4174, Online. Association for Computational Linguistics.

Yoon Kim and Alexander M. Rush. 2016. Sequence-level knowledge distillation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1317–1327, Austin, Texas. 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 *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

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.

Fengfu Li, Bo Zhang, and Bin Liu. 2016. Ternary weight networks. In *Proceedings of 1st International NIPS Workshop on EMDNN*.

Zheng Li and Christopher De Sa. 2019. Dimension-free bounds for low-precision training. 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 11728–11738.

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.

Zhouhan Lin, Matthieu Courbariaux, Roland Memisevic, and Yoshua Bengio. 2016. Neural networks with few multiplications. In *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*.

Yinhan Liu, Jiatao Gu, Naman Goyal, Xian Li, Sergey Edunov, Marjan Ghazvininejad, Mike Lewis, and Luke Zettlemoyer. 2020. Multilingual denoising pre-training for neural machine translation. *Transactions of the Association for Computational Linguistics*, 8:726–742.

Xindian Ma, Peng Zhang, Shuai Zhang, Nan Duan, Yuexian Hou, Ming Zhou, and Dawei Song. 2019. A tensorized transformer for language modeling. 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 2229–2239.

Paul Michel, Omer Levy, and Graham Neubig. 2019. Are sixteen heads really better than one? 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 14014–14024.

Ramesh Nallapati, Bowen Zhou, Cicero dos Santos, Çağlar Gulçehre, and Bing Xiang. 2016. Abstractive text summarization using sequence-to-sequence rnns and beyond. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, pages 280–290.

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.

Gabriele Prato, Ella Charlaix, and Mehdi Rezagholizadeh. 2020. Fully quantized transformer for machine translation. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1–14, 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.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version

of bert: smaller, faster, cheaper and lighter. *ArXiv preprint*, abs/1910.01108.

Abigail See, Peter J. Liu, and Christopher D. Manning. 2017. Get to the point: Summarization with pointer-generator networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1073–1083, Vancouver, Canada. Association for Computational Linguistics.

Sam Shleifer and Alexander M Rush. 2020. Pre-trained summarization distillation. *ArXiv preprint*, abs/2010.13002.

Linting Xue, Noah Constant, Adam Roberts, Mihir Kale, Rami Al-Rfou, Aditya Siddhant, Aditya Barua, and Colin Raffel. 2021. mT5: A massively multilingual pre-trained text-to-text transformer. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 483–498, Online. Association for Computational Linguistics.

Ofir Zafrir, Guy Boudoukh, Peter Izsak, and Moshe Wasserblat. 2019. Q8bert: Quantized 8bit bert. In *2019 Fifth Workshop on Energy Efficient Machine Learning and Cognitive Computing-NeurIPS Edition (EMC2-NIPS)*, pages 36–39. IEEE.

Wei Zhang, Lu Hou, Yichun Yin, Lifeng Shang, Xiao Chen, Xin Jiang, and Qun Liu. 2020. TernaryBERT: Distillation-aware ultra-low bit BERT. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 509–521, Online. Association for Computational Linguistics.

Ben Zhou, Kyle Richardson, Qiang Ning, Tushar Khot, Ashish Sabharwal, and Dan Roth. 2021. Temporal reasoning on implicit events from distant supervision. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1361–1371, Online. Association for Computational Linguistics.

A Details of TWN Quantization

When quantizing using 2 bits (which is also know as ternarization), following Zhang et al. (2020), we apply the TWN method (Li et al., 2016). To quantize $\mathbf{w}$, we are looking for scaling factor $\alpha > 0$ and $\mathbf{b} \in \{-1, 0, 1\}^n$ such that $\mathbf{w} \sim \alpha\mathbf{b}$ where n is the dimension of $\mathbf{w}$. To minimize the quantization error, we have the following optimization problem:

$$\alpha^*, \mathbf{b}^* = \arg\max_{\alpha,\mathbf{b}} ||\mathbf{w} - \alpha\mathbf{b}||^2$$

$$\text{where } \alpha > 0, \mathbf{b} \in \{-1, 0, 1\}^{dim(\mathbf{w})}$$

Denote Δ as a threshold and $I_\Delta(x)$ be a function such that

$$I_\Delta(x) = \begin{cases} 1, \text{ if } x > \Delta \\ 0, \text{ if } -\Delta \le x \le \Delta \\ -1, \text{ if } x < -\Delta \end{cases}$$

and denote set $J_\Delta = \{i \mid I_\Delta(\mathbf{w}_i) \neq 0\}$, then according to Hou and Kwok (2018), the solution to the previous optimization problem can be reached at

$$\mathbf{b}^* = I_{\Delta^*}(\mathbf{w}), \alpha^* = \frac{||\mathbf{w} \odot \mathbf{b}^*||_1}{||\mathbf{b}^*||_1},$$

$$\text{with } \Delta^* = \arg\max_{\Delta} \frac{1}{|J_\Delta|} \left(\sum_{i \in J_\Delta} |\mathbf{w}_i| \right)$$

where $\odot$ is element-wise multiplication and $||\cdot||_1$ is the l_1-norm. To approximate this result, we set $\Delta^* = 0.7||\mathbf{w}||_1/dim(\mathbf{w})$ then compute α^* and $\mathbf{b}^*$ accordingly.

B Details of Distillation Losses

The distillation losses is defined as the following:

$$\mathcal{L}_{\text{dist}} = \mathcal{L}_{\text{logits}} + \mathcal{L}_{\text{att}} + \mathcal{L}_{\text{hid}}$$

In this section we'll go through each part of the losses. We denote $\phi_{enc}(\cdot), \phi_{dec}(\cdot)$ as the functions that map the index of an encoder/decoder layer of the student model to the index of the teacher model layer that it is trained to emulate, the details of which is discussed in §2.2, and we use l^S_{enc}, l^S_{dec} to denote the number of encoder layers and decoder layers of the student model. To illustrate, if $l^S_{enc} = 3, l^S_{dec} = 2$, we would have:

$$\phi_{enc}(0, 1, 2) = 0, 3, 5, \quad \phi_{dec}(0, 1) = 0, 5$$

For simplicity, we use superscript $\cdot^S, \cdot^T$ to distinguish counterparts from the student model and teacher model respectively.

Next, we will explain the definition of each part of the distillation losses.

Firstly, $\mathcal{L}_{\text{logits}}$ is the Mean Squared Error (MSE) between the output logits of the student model and that of the teacher model, i.e.

$$\mathcal{L}_{\text{logits}} = MSE(logits^S, logits^T)$$

Secondly, $\mathcal{L}_{\text{att}}$ is the attention distillation loss, which is the sum of distillation losses of encoder

attentions (EA), decoder attentions (DA), and cross attention (CA), i.e.

$$\mathcal{L}_{\text{att}} = \mathcal{L}_{\text{EA}} + \mathcal{L}_{\text{DA}} + \mathcal{L}_{\text{CA}}$$

where

$$\mathcal{L}_{\text{EA}} = \sum_{i=1}^{l^S_{enc}} MSE(EA^S_i, EA^T_{\phi_{enc}(i)})$$

$$\mathcal{L}_{\text{DA}} = \sum_{i=1}^{l^S_{dec}} MSE(DA^S_i, DA^T_{\phi_{dec}(i)})$$

$$\mathcal{L}_{\text{CA}} = \sum_{i=1}^{l^S_{dec}} MSE(CA^S_i, CA^T_{\phi_{dec}(i)})$$

with the subscripts $i, \phi(i)$ specifying the indices of the layers.

Finally, $\mathcal{L}_{\text{hid}}$ is the distillation loss between all the hidden states between student layers and teacher layers, which include encoder hidden states (EHS) and decoder hidden states (DHS):

$$\mathcal{L}_{\text{hid}} = \mathcal{L}_{\text{EHS}} + \mathcal{L}_{\text{DHS}}$$

where

$$\mathcal{L}_{\text{EHS}} = \sum_{i=1}^{l^S_{enc}} MSE(EHS^S_i, EHS^T_{\phi_{enc}(i)})$$

$$\mathcal{L}_{\text{DHS}} = \sum_{i=1}^{l^S_{dec}} MSE(DHS^S_i, DHS^T_{\phi_{dec}(i)})$$

C Dataset Statistics

Dataset	Dataset Split Count			Mean Token Length	
	Train	Valid.	Test	Source	Target
CNN/DM	87,113	13,368	11,490	691	52
XSUM	204,045	11,332	11,334	374	21
ELI5	272,634	9,812	24,512	Q: 38 D: 1,672	111
WMT16 En-Ro	610,320	1,999	1,999	21	21

Table 4: Dataset Statistics.

D ELI5 Additional Details

In this section, we present additional details for the ELI5 dataset.

D.1 Dense Retriever

We were not able to find a public version of supporting documents for ELI5, and thus followed the author's implementation[4] to train a dense retriever that retrieves support documents from Wikipedia. Our trained retriever achieves a similar performance compared with the one reported in the author's implementation (recall: ours 0.3273, reported 0.3247).

D.2 Evaluating ELI5 Results

We use the ROUGE-SCORE package[5] to calculate rouge scores through the paper. However, as the author of ELI5 pointed out[4], the original rouge implementation used in ELI5 and BART papers performs additional normalization. For consistency, we also reported results for ELI5 using the same ROUGE-SCORE package, which differs from the one used in ELI5/BART. Here we compared the performance of our trained ELI5 baseline model with the public one using the rouge implementation used in ELI5/BART papers.

Model	Rouge Setting	R1	R2	RL
BART-base (ours)	rouge-score	26.02	5.11	15.36
	BART/ELI5	29.19	5.59	25.88
BART-large reported (Lewis et al., 2020)	BART/ELI5	30.60	6.20	24.30

Table 5: Comparison when using different rouge implementation.

Results in Table 5 shows that the performance of our base-size model is close to the one with large-size reported in Lewis et al. (2020). This signals that our baseline model for ELI5 is well-trained.

E Visualizations of Experimental Results on XSUM and ELI5 datasets

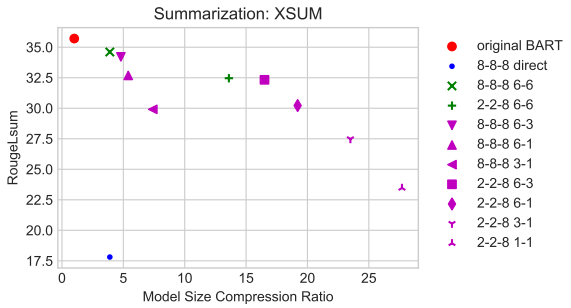

Figure 2: Visualization of performance v.s. model footprint compression ratio on XSUM based on Table 1.

[4] https://yjernite.github.io/lfqa.html

[5] https://pypi.org/project/rouge-score/

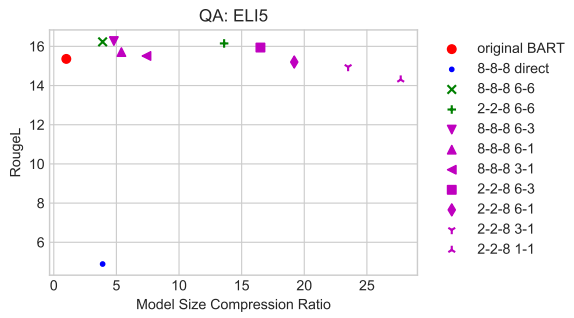

Figure 3: Visualization of performance v.s. model footprint compression ratio on ELI5 based on Table 1.

F Comparisons on "Shrink and Finetune"

We benchmarked the performance of three randomly picked models with the "Shrink and Finetune" schema proposed in Shleifer and Rush (2020). We ran the models using the same hyperparameter settings we used in this paper. The results are shown in Table 6.

We found that when using distillation losses between the teacher and the student, the performance are slightly better than the "Shrink and Finetune" method under our setting. This signals that having guidance in weighting is important for a quantized and distilled model to learn well.

Model	Loss	R1	R2	RL
CNN/DM	Ours	**42.52**	**20.04**	**40.05**
8-8-8 6-1	S&F	42.29	19.92	39.83
XSUM	Ours	**40.06**	**17.34**	**32.46**
2-2-8 6-6	S&F	39.69	17.27	32.27
ELI5	Ours	**27.10**	**5.15**	**16.23**
8-8-8 6-6	S&F	26.95	5.10	16.16

Table 6: Performance comparison between the loss used in this paper and the "shrink and finetune" loss from (Shleifer and Rush, 2020).