machine learning & knowledge extraction

MDPI

Article

Efficient Latent Space Compression for Lightning-Fast Fine-Tuning and Inference of Transformer-Based Models †

Ala Alam Falaki * and Robin Gras

School of Computer Science, University of Windsor, Windsor, ON N9B 3P4, Canada

* Correspondence: alamfal@uwindsor.ca

† This paper is an extended version of our paper published in Falaki, A.A.; Gras, R. A Robust Approach to Fine-Tune Pre-Trained Transformer-Based models for Text Summarization through Latent Space Compression. In Proceedings of the 2022 21st IEEE International Conference on Machine Learning and Applications (ICMLA), Nassau, Bahamas, 12–14 December 2022.

Abstract: This paper presents a technique to reduce the number of parameters in a transformer-based encoder–decoder architecture by incorporating autoencoders. To discover the optimal compression, we trained different autoencoders on the embedding space (encoder's output) of several pre-trained models. The experiments reveal that reducing the embedding size has the potential to dramatically decrease the GPU memory usage while speeding up the inference process. The proposed architecture was included in the BART model and tested for summarization, translation, and classification tasks. The summarization results show that a 60% decoder size reduction (from 96 M to 40 M parameters) will make the inference twice as fast and use less than half of GPU memory during fine-tuning process with only a 4.5% drop in R-1 score. The same trend is visible for translation and partially for classification tasks. Our approach reduces the GPU memory usage and processing time of large-scale sequence-to-sequence models for fine-tuning and inference. The implementation and checkpoints are available on GitHub.

Keywords: transformers; autoencoder (AE); sequence-to-sequence (seq2seq); compression; summarization; translation; classification

Citation: Alam Falaki, A.; Gras, R. Efficient Latent Space Compression for Lightning-Fast Fine-Tuning and Inference of Transformer-Based Models. *Mach. Learn. Knowl. Extr.* **2023**, *5*, 847–867. https://doi.org/10.3390/make5030045

Academic Editors: Moamar Sayed-Mouchaweh, Mohd Arif Wani, Vasile Palade and Mehmed M. Kantardzic

Received: 15 June 2023
Revised: 14 July 2023
Accepted: 26 July 2023
Published: 30 July 2023

1. Introduction

This paper is an extended version of the work published in [1] to explore the proposed architecture in more detail to further study its strength and shortcomings. The primary focus of this paper is to create a more efficient fine-tuning paradigm for pre-trained transformer-based [2] models. It is no secret that the current landscape in natural language processing (NLP) portrays a direct correlation between the model's scale and capabilities. Admittedly, the trend is more visible in decoder-only large language models like GPT-family [3–5] and LLaMA [6]. However, other task-specific models can also experience a boost in performance with a higher number of parameters. This tendency has serious environmental [7] and usability impacts.

Exploring the new possibilities to enhance the accessibility of large-scale models is a crucial research area. The performance of an architecture, no matter how capable it may be, becomes irrelevant if researchers cannot make further improvements or practitioners cannot build upon it. Even the process of loading some of the large-scale models, like the encoder-only switch transformer [8], necessitates a substantial amount of engineering effort. The impact of this trend on the sequence-to-sequence (seq2seq) architecture is even more pronounced due to the presence of both an encoder and a decoder component. Without adequate resources, it is challenging to effectively utilize well-known seq2seq models such as BART [9] and Pegasus [10], which consist of 400 M and 570 M parameters, respectively. It is worth noting that these numbers are considered relatively small compared to the current standards.

This research examines how the dimensionality reduction capability of autoencoders [11] can be used to compress the latent space of an encoder–decoder architecture. Reducing the size of the embedding space leads to a smaller decoder, as it reduces the input dimension of the decoder. Both the fine-tuning and inference processes are expected to benefit from this approach. However, we anticipate a significant improvement during inference, as the decoder plays a crucial role in this stage. The main contributions of this paper are as follows:

- The best algorithm and architecture to perform an effective embedding space compression were found.
- An extensive number of experiments were conducted to find the perfect balance between the autoencoder's compression rate and the model's capability.
- The proposed architecture underwent testing across various datasets and was evaluated for summarization, translation, and classification tasks.
- The autoencoder's capacity to effectively generalize unseen datasets and diverse tasks was explored.

The rest of the paper is organized as follows. In the next section, we provide an overview of the prior literature on the topic of model compression. Section 3 will delve into a comprehensive explanation of the proposed architecture and its individual components, offering a more detailed perspective. Next, we discuss the experiments that were carried out to both validate the approach and assess its effectiveness across various tasks. Subsequently, we present the outcomes of the experiments, concluding with a comprehensive analysis of the findings. You can find the implementation details and pre-trained weights in the provided GitHub repository: https://github.com/AlaFalaki/compressedBART (accessed on 13 July 2023).

2. Background

The transformer architecture has established its dominance across various NLP tasks. The BART [9] model reigned supreme on the summarization task leaderboard for a long time and demonstrated proficiency in translation as well. A number of studies extend it for summarization like the current state-of-the-art approach, BRIO [12], which generates a large pool of summaries and select the best candidate, or the HydraSum [13] architecture, with multiple decoders for capturing different styles.

By pre-training on a multilingual dataset, the mBART [14] model has achieved an enhanced performance in machine translation task. Various approaches have been investigated to enhance translation quality, including the utilization of back translation for monolingual models [15] and parameter sharing across layers to reduce the number of parameters and computational time [16].

For several classification tasks, the pre-trained XLNet [17] model demonstrated superior performance compared to BERT. LadaBERT [18] presents a novel approach that combines various dimensionality reduction techniques, including low rank decomposition and pruning. This approach leads to a compact and highly efficient model that can seamlessly integrate into diverse applications. The ERNIE-Doc [19] model introduces a recursive architecture that offers an improved efficiency in handling large sequences. It surpasses competing models in multiple tasks, including classification.

Additional researchers are actively investigating the pre-training of large models to achieve high performance across multiple downstream tasks. The T5 [20] architecture is one of the well-known examples that resulted in in a pre-trained model to handle translation, summarization, question answering, and classification with high accuracy. These advancements are scaling up the parameter count and slowing the inference process.

Multiple research studies have already been conducted to tackle the growing size of neural network models. Quantization [21] is one of the first experiments which apply to any deep learning model by using a half-precision (16-bit) floating point to greatly reduce the network size and memory usage. Micikevicius et al. proposed a mixed precision algorithm in [22] to further close the gap in evaluation results. Recent works experimented with the effect of the knowledge distillation [23] method to transfer information from a larger

network to a smaller one without a significant loss in accuracy. There are multiple papers that present different combinations of fine-tuning and distillation on top of BERT [24,25]. However, DistilBERT [26] obtained the best results by training the smaller student model on BERT and then fine-tuning it for downstream tasks, resulting in a more generalized pre-trained model. Lastly, the DistilBART [27] model aimed to create a smaller student network for summarization.

The pruning [28] method's influence on transfer learning has recently gained attention from researchers. It refers to determining the parts in the network that have the weaker effect on the model accuracy and removing them without compromising the model's accuracy on downstream tasks. The main ideas are to either focus on finding the less important weights [29] or components such as the number of self-attention heads [30] and layers [31]. This technique was also used in the lottery ticket hypothesis [32,33] to uncover subnetworks performing on par with the full model. This method focusing on linear layers is adopted for the seq2seq architectures like BART and Pegasus to speed up the inference process [34].

The latest research area focuses on rethinking the self-attention mechanism to eliminate its quadratic memory usage connection with respect to the input sequence length. The goal is to find the best trade-off between performance and memory usage. Big Bird [35] and Longformer [36] papers experiment on different attention patterns to reduce connections and result in fewer computations. Wang et al. presented the Linformer [37] network that projects the self-attention vectors to lower dimensions. Reformer [38] paper studies the idea of grouping key and query vectors based on locality-sensitive hashing (LSH) to reduce the computations needed to find similar vectors.

It is important to highlight several studies that have effectively combined two or more methods to construct smaller models, while maintaining a high level of accuracy without significant compromise. The literature contains numerous experiments that explore the combination of distillation with pruning [39] or quantization [40], among others. Furthermore, Thierry et al. made the EdgeBERT [41] model by leveraging both pruning and quantization along with other methods. These techniques are not exclusive, and it is possible to study them independently without testing all possible combinations. This is why we can focus our study only on one reduction method and only a few important models.

3. Proposed Method

Our proposed architecture is a sequence-to-sequence transformer (T) model that uses a pre-trained autoencoder connecting the network's encoder (T_{enc}) to its decoder (T_{dec}) (Figure 1). The mentioned approach will result in a smaller T_{dec} and reduce the overall number of trainable parameters. The modules are described in the following sub-sections.

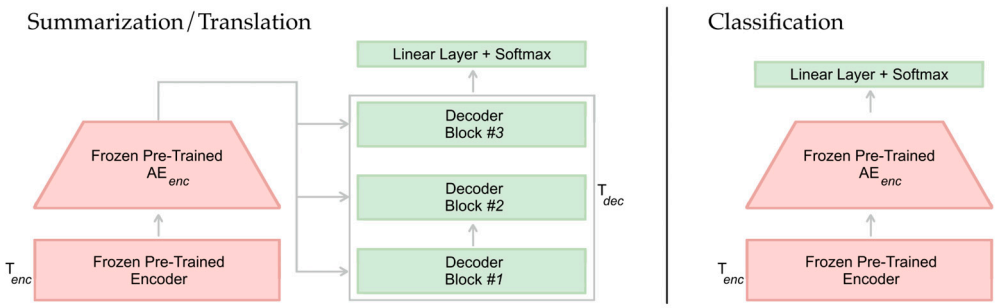

Figure 1. The proposed architecture in encoder–decoder (left) and encoder-only (right) setting. The diagram's red-colored components indicate being pre-trained and frozen during the training of the summarizer model. The green-colored units are learned from scratch for the summarization task.

3.1. Transformer Encoder (T_{enc})

The encoder (T_{enc}) part of the proposed architecture is a pre-trained transformer-based model. We have selected a set of models that include some of the best models for text summarization: BERT, DistilBERT, BART_{enc} (only encoder), and a custom transformer model as a baseline comparison. The custom transformer model is used to evaluate the effectiveness of our approach on a small pre-trained model with only six encoder layers. Subsequently, its scores are used as a baseline and are not supposed to be competitive with the ones of the other approaches. This component is frozen during the training of the summarizer model.

3.2. Autoencoder (AE)

The *AE* (Figure 2) purpose is to reduce the T_{enc}'s output size to a smaller latent space using the following equation:

$$X'_{AE} = AE_{dec}(AE_{enc}(X_{AE}))Z = AE_{enc}(X_{AE}) \quad (1)$$

where $X_{AE}^{S\times D}$ is the input, AE_{enc} indicates the encoder responsible for compressing the input to latent space $Z^{S\times C}$ and a decoder AE_{dec} generating the output $X\prime_{AE}^{S\times D}$, trying to reconstruct the input $X_{AE}^{S\times D}$ during the training process. Variables S and D denote the sequence length and input's embedding dimension, respectively. The values depend on the chosen pre-trained model's configuration and are set to 512 and 768 in the experiments. However, C, representing the compressed latent space size, will vary to find out the optimal latent space size. A six-layer linear *AE* (three for encoder and three for decoder) is selected after comparing its reconstruction ability to the same architecture with long short-term memory (LSTM) [42] or convolutional neural network (CNN) [43] building blocks (see Section 5.1).

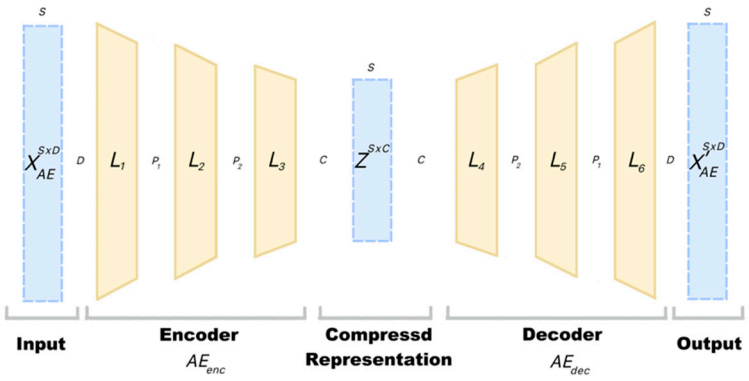

Figure 2. The linear autoencoder architecture with three encoder layers (L1, L2, L3) and three decoder layers (L4, L5, L6). Tensors X, X', and Z represent the input, output, and compressed latent representation, respectively. The autoencoder maintains the same sequence length (S) during compression and only reduces the embedding size (D).

The final *AE* architecture with six linear layers is independently trained for each selected pre-trained encoder T_{enc}. It attempts to reconstruct the output of T_{enc} using a smaller representation Z. The frozen AE_{enc} is then used in our summarizer architectures to pass a compressed representation to the decoder T_{dec} (from size D to C). Refer to Appendix A for more information about the autoencoder architecture details.

3.3. Transformer Decoder (T_{dec})

The decoder component of the architecture (T_{enc}) is an original transformer decoder with three layers and a linear head on top in all the experiments. It is the only piece of the network that is not frozen after the *AE* has been trained. Its embedding dimension ties to

the *AE*'s latent space size (C) that can drastically alter the architecture's overall number of trainable parameters.

4. Experiments

We performed two experiments to: *(i)* evaluate the effectiveness of the proposed architecture as described in Section 3 and *(ii)* showcase its effectiveness by integrating it with BART-base model on summarization, translation, and classification tasks.

The first study (we call it "validation") starts by justifying the use of the linear layer autoencoder by comparing its performance to compress and reconstruct the BERT's output with a different number of layers and building units (LSTM and CNN). Then, we investigated the advantage of using "frozen autoencoders" with several studies listed below:

- AE (presented architecture): A pre-trained and frozen autoencoder that connects the encoder to a 3-layer decoder.
- AE-S: The same architecture without pre-training the autoencoders. They will be trained jointly with the decoder from scratch.
- LL: A small 1-layer learnable linear model to lower the encoder's output dimensionality from D to C.
- PCA: The classical dimensionality reduction algorithm, incremental PCA [44], trained to project the outputs of the encoder to the 458 first principal components to preserve more than 90% of variances and use them as the decoder input.

The summaries are generated using three different decoding strategies. The fastest and easiest method is selecting the most probable output at each timestep, known as the greedy algorithm. A more extensive approach is to use the beam search algorithm that develops K paths in parallel using the top K tokens with the highest scores. Finally, we used the weighted random sampling algorithm that randomly chooses a token from the top K probable outputs proportionally to their respective probabilities.

Following the "Validation" phase, we thoroughly explore the potential of the proposed architecture by evaluating it on different tasks. The compression rates are slightly different in these experiments because the compression sizes (latent space dimension) must be dividable by the number of the decoder's attention heads (12).

Evaluation: The summarization and translation experiments use ROUGE [45] and BLEU [46] scores to measure the model's generation quality. The ROUGE score is the standard metric to calculate how close a generated summary is, compared to the target. It will measure the overlapping n-grams between the two (R-1 for unigrams, R-2 for bigrams, etc.) and the longest common subsequence n-grams (R-L). The classification task is measured using the accuracy metric. The BLEU score compares the machine-generated translation with one or more reference translations. It counts how many words and phrases from the machine translation are also present in the references and assigns a score based on this overlap. The sentient analysis study will use the BART architecture with a classification head.

Hardware and Hyperparameters: The effects of the proposed architecture will be analyzed in subsections aimed at the GPU memory consumption and fine-tuning/inference computational time. The experiments aim for maximum performance using Nvidia V100 GPU with 32 GB memory. Therefore, each task uses the largest possible batch size with respect to hardware constraints. Moreover, the datasets are vastly varied in size and characteristics. Thus, it is not possible to compare the tasks against each other.

Lastly, the autoencoder models are trained with fixed hyperparameters to make them comparable. They are trained using the 1cycle [47] training policy that increases the learning rate from 2×10^{-5} to a maximum of 5×10^{-4}, while decreasing the momentum beta from 0.95 to 0.85 for a faster convergence. We also used the Adam [48] optimizer and the label smoothing CrossEntropy [49] loss function for both autoencoder and summarizer models.

Datasets: The CNN/Dailymail (300 K samples) [50] and the Newsroom (1.3 M samples) [51] datasets were used in the following experiments. We combined the training set of

mentioned datasets and randomly selected 60% of the samples to train the autoencoders. This was performed to hold onto unseen data to evaluate the generalization ability of the summarizer model. The summarizer models are trained using the CNN/Dailymail training set and evaluated on the pre-defined test set.

The translation task utilizes the WMT16 dataset's Romanian-to-English subset [52]. It has 610 K samples in the training set and 2 K for each validation and test set. The training set is used to create a Romanian language tokenizer with the same vocabulary size as BART's pre-trained tokenizer.

The Sentiment140 [53] dataset is used for the classification experiment. It is a sentiment analysis dataset consisting of twitter data with corresponding annotations (0 for negative and 4 representing positive). There are 1.6 M data points with random 80–10–10 splits for training, validation, and testing, respectively. A simple preprocessing step was performed to remove unnecessary characters (e.g., @ signs, URLs, or HTML codes).

5. Results

In the upcoming subsections, we will initially conduct a comprehensive examination of the architecture's viability and subsequently assess its performance in summarization, translation, and classification tasks.

5.1. Validation

To assess their performance, we conducted experiments involving various types and depths of AE layers. Specifically, we trained autoencoder architectures using linear, long short-term memory (LSTM), and convolutional neural network (CNN) layer types, with the mean square error (MSE) loss, over a span of seven epochs. A latent space representation of size 64 was selected for this comparison benchmark (Table 1). The linear autoencoder demonstrates a superior performance compared to both LSTM and CNN architectures (refer to Appendix A, Table A1 for the complete list of all compression rates). Also, the six-layer design choice results in a better score in all experiments.

Table 1. The comparison of MSE loss between linear, LSTM, and CNN blocks to train an autoencoder with a 64 compression size.

Number of Layers Types	4	6	8
Linear	0.0813	0.0766	0.0776
LSTM	0.0863	0.0810	0.0849
CNN	0.2666	0.2659	0.2750

Tables 2 and 3 present the comparisons of model sizes in terms of the number of parameters. The first step is calculating the decoder's size without using the dimensionality reduction method. As previously mentioned, the default encoder's representation dimension (d_{model}) is 768 (based on the choice of the encoder), which results in a decoder with either 48 M (for BART_{enc}) or 33 M (other options) parameters. We utilize this value as a reference number to calculate the reduction percentage. It is important to keep in mind that even though there are parameters being added to the model from the autoencoder that affects the percentage, the number of trainable parameters in all the experiments is equal to the decoder size since the rest of the components (encoder and autoencoder) are frozen during the tests. We have decoders with different parameter counts while using the same hyper-parameters, because we use each model's pre-trained tokenizers with different vocabulary sizes, which affects the decoder's last layer output dimension. BERT, DistilBERT, and transformer models use BERT's pre-trained tokenizer, and BART uses its own.

Table 2. Comparing the decoder's number of parameters in a 3-layer decoder network with a 768 input size to the number of parameters of the same decoder after applying the autoencoder to reduce the encoder's output dimension.

Decoder Input Dimension	AE Parameters Count	Decoder Parameters Count (by Encoder Type)			
		(Distil)BERT/ Transformer	Reduction (%)	$BART_{enc}$	Reduction (%)
768 (Default)	-	32 M	-	48 M	-
C = 512	2.3 M	21 M	26.26	32 M	28.48
C = 384	1.8 M	16 M	44.44	24 M	46.18
C = 128	1.1 M	5 M	79.84	8 M	80.91
C = 32	1 M	1 M	92.47	2 M	93.5

Table 3. Comparing the total number of parameters in a 3-layer decoder network with a 768 input size to the number of parameters of the same decoder after applying the autoencoder to reduce the encoder's output dimension.

Decoder Input Dimension	AE Parameters Count	Network's Total Parameters Count (by Encoder Type)			
		Transformer	BART	$BERT_{enc}$	DistilBERT
768 (Default)	-	70 M	188 M	142 M	98 M
C = 512	2.3 M	61 M	174 M	134 M	90 M
C = 384	1.8 M	55 M	165 M	128 M	84 M
C = 128	1.1 M	44 M	149 M	116	75 M
C = 32	1 M	40 M	143 M	112 M	68 M

Table 4 displays the results of the proposed method on the text summarization task, with the utilization of the classic ROUGE (R-1, R-2, and R-L) metric, employing the greedy decoding strategy (refer to Appendix B for the weighted random sampling and beam search scores). Autoencoders are trained with four (32, 128, 384, 512) different latent space dimensions (C) to analyze the effect of these compression rates. The models are also evaluated using the BERTScore [54] metric, and the results follow the same trend (refer to Appendix C).

In addition to reducing the number of parameters in the network by decreasing the decoder size, our configurations exhibit the capability to enhance the model's summarization ability, resulting in higher scores compared to the original setup for both the transformer and DistilBERT models. The experiments show that adding an AE with $C = 512$ outperforms the same vanilla encoder–decoder network. Both $BART_{enc}$ and BERT experiments with the same compression rate outperformed their vanilla model in several metrics by only using the beam search method. At the same time, the rest of the scores with a smaller decoder are still competitive. An even more interesting result is that a smaller dedicated transformer with the proposed method and $C = 384$ performed better than both $BART_{enc}$ and BERT in the summarization benchmark.

Table 5 presents the results of our ablation study, illustrating the impact of the autoencoder pre-training step on the final score. It surpasses both training the AE_{enc} jointly with the network from scratch (AE S) and using a simple learnable linear layer for projecting (LL). Lastly, the PCA dimensionality reduction technique does not produce desirable results as well. The results also show that using an autoencoder with a latent space size of 384 ($C = 384$) results in a ROUGE score close to the vanilla model. It reduces the decoder size by 46% from 48 M to 24 M for the $BART_{enc}$ model and 44% from 33 M to 16 M parameters for the other models. The critical point is that the combination of this configuration in association with the greedy decoding algorithm shows no noticeable degrading quality in the generated summary (refer to Appendix D).

Table 4. The ROUGE score for using a pre-trained autoencoder on top of pre-trained transformer-based encoders with different compression sizes. The summaries are generated using greedy decoding method. The percentages indicate the relative change in the ROUGE score compared to the baselines.

Models	R-1	R-2	R-L
Transformer	0.346	0.143	0.312
+AE ($C = 512$)	0.368 (106%)	0.157 (109%)	0.325 (104%)
+AE ($C = 384$)	0.363 (104%)	0.154 (107%)	0.322 (103%)
+AE ($C = 128$)	0.308 (89%)	0.114 (79%)	0.286 (91%)
+AE ($C = 32$)	0.156 (45%)	0.019 (13%)	0.174 (55%)
BART_{enc}	0.355	0.142	0.310
+AE ($C = 512$)	0.341 (96%)	0.128 (90%)	0.298 (96%)
+AE ($C = 384$)	0.332 (93%)	0.121 (85%)	0.291 (93%)
+AE ($C = 128$)	0.257 (72%)	0.063 (44%)	0.239 (77%)
+AE ($C = 32$)	0.145 (40%)	0.014 (9%)	0.168 (54%)
BERT	0.349	0.133	0.306
+AE ($C = 512$)	0.339 (97%)	0.123 (92%)	0.298 (97%)
+AE ($C = 384$)	0.332 (95%)	0.119 (89%)	0.294 (96%)
+AE ($C = 128$)	0.278 (79%)	0.074 (55%)	0.256 (83%)
+AE ($C = 32$)	0.168 (48%)	0.021 (15%)	0.187 (61%)
DistilBERT	0.317	0.124	0.283
+AE ($C = 512$)	0.333 (105%)	0.123 (99%)	0.298 (105%)
+AE ($C = 384$)	0.334 (105%)	0.122 (98%)	0.297 (104%)
+AE ($C = 128$)	0.287 (90%)	0.083 (66%)	0.265 (93%)
+AE ($C = 32$)	0.161 (50%)	0.020 (16%)	0.180 (63%)

Table 5. The comparison between using the pre-trained autoencoder (AE), training the autoencoder's encoder jointly with the network from scratch (AE S), using a simple linear layer model for the projection (LL), and PCA to perform the dimensionality reduction. The percentages indicate the relative change in the ROUGE score compared to the baseline (BERT).

Models	R-1	R-2	R-L
BERT	0.349	0.133	0.306
+AE ($C = 512$)	0.339 (97%)	0.123 (92%)	0.298 (97%)
+AE S ($C = 512$)	0.289 (82%)	0.079 (59%)	0.262 (85%)
+LL ($C = 512$)	0.277 (79%)	0.083 (62%)	0.260 (84%)
+PCA ($C = 458$)	0.143 (40%)	0.016 (12%)	0.156 (50%)

Our findings demonstrate that by reducing the encoder's output dimension from 768 to 384, we achieve a 44% reduction in decoder size while preserving 95% of the R-1 score in the worst-case scenario using BERT as the encoder. (refer to Appendix E for additional experiments on intermediate compression rates) Alternatively, with DistilBERT, we observe a potential increase of up to 105% in the R-1 score. Moreover, even with an almost 80% reduction in the decoder size, we can still achieve 90% of the ROUGE score with our dedicated transformer and DistilBERT architectures. This method will be integrated into a full pre-trained sequence-to-sequence model to measure its footprint in the next section.

5.2. Summarization

This section's experiment utilizes the proposed architecture in a sequence-to-sequence pre-trained model (BART) to demonstrate its abilities. We fine-tune (i) vanilla BART-base and (ii) BART-base with autoencoder (various compression rates), while both BART's encoder and the AE_{enc} are frozen. The ROUGE score for these experiments (using beam search decoding) and the number of decoder parameters are reported in Table 6. The decoder component is responsible for generating tokens one at a time during inference, so compressing it will largely optimize the generation process.

Table 6. The ROUGE score results from integrating BART-base with our proposed architecture for summarization. The autoencoders are frozen, and "fe" stands for "frozen encoder". The last column indicates the number of parameters in the decoder part of the network. The summaries are generated using the beam search decoding method.

Models		R-1	R-2	R-L	# Dec Params
$BART_{fe}$	BART	0.419	0.198	0.393	96 M
	+AE (C = 504)	0.401	0.182	0.375	56 M
	+AE (C = 384)	0.400	0.181	0.374	40 M
	+AE (C = 120)	0.351	0.133	0.331	21 M
	+AE (C = 24)	0.182	0.026	0.170	11 M

Our proposed architecture with a 384 compression size shows a 4.5% drop in the R-1 score while reducing the model's decoder size by nearly 60%. The slight decrease in the ROUGE score does not seem to translate to a lower-quality summary (refer to Appendix F). Moreover, the network size decrease will compensate for a few lost overlapping n-grams in summaries by reducing the computational time and memory usage by a large margin at training and inference time. The following experiments are performed using a batch size of 16 and 32 for training and inference, respectively.

5.2.1. Computational Time

As stated in Table 7, the BART + AE ($C = 384$) architecture can be fine-tuned 45% faster. It is a one-hour reduction (or 45%) compared to the BART's 2:40 h for one epoch. Moreover, the inference (beam search) timing is a more important factor since the models are usually fine-tuned once to be used in numerous applications in the future. The 54% decrease in inference time on the test set (the 5:09 compared to 2:20 h) could greatly impact the responsiveness of downstream applications.

Table 7. Comparing vanilla BART and the proposed architecture on the minutes it takes to fine-tune the model for 1 epoch and performing inference on the full test set (summarization task).

Model	Computational Time (Minutes)	
	Fine-Tuning	Inference
BART	145	309
BART + AE (C = 384)	79	140

The block pruning technique [34] reported a 1.35 speedup by reducing the "linear" parameter count from 99 M to 23 M in the BART-base model. The smallest DistilBART [27] model showed a 1.66 faster inference rate with a 100 M decoder size. Our approach resulted in a much faster inference time with a 2.2 increase in speed rate.

5.2.2. GPU Memory Usage

The comparison of GPU memory consumption is shown in Table 8. The BART+AE ($C = 384$) experiment shows a 57% reduction in GPU memory usage while fine-tuning and a 20% reduction during inference. The massive drop in GPU utilization while fine-tuning makes it easier for both researchers and practitioners to try larger models in their experiments with minimal sacrifice in the quality.

Table 8. Comparing vanilla BART and the proposed architecture on the GPU memory (MBs) used to fine-tune the model for 1 epoch and performing inference on the full test set (summarization task).

Model	GPU Memory Usage (MB)	
	Fine-Tuning	Inference
BART	28,384	30,624
BART + AE (C = 384)	10,240	23,680

5.3. Translation

The BART-base model is fine-tuned for translation to set a baseline score. Additionally, the proposed architecture is fine-tuned on the WMT16 dataset, as described in Section 3. The translations are generated using beam search (K = 4), and the BLEU score is used to measure the translation's quality, as presented in Table 9. The proposed architecture with a 50% compression rate (from 768 to 384) shows a ~10% decrease in the BLEU score. Meanwhile, the generated translations are close in meaning, even identical in some cases (refer to Appendix G). This experiment also reveals that an autoencoder pre-trained on one dataset can perform well on others without additional fine-tuning. The following experiments are conducted using a batch size of 32 and 64 for training and inference, respectively.

Table 9. The BLEU score results from integrating BART-base with our proposed architecture for translation. The autoencoders are frozen, and "fe" stands for "frozen encoder". The last column indicates the number of parameters in the decoder part of the network. The translations are generated using the beam search decoding method.

Models		BLEU	# Dec Params
BART		21.05	96 M
$BART_{fe}$	+AE (C = 504)	18.93	56 M
	+AE (C = 384)	18.63	40 M
	+AE (C = 120)	13.95	21 M
	+AE (C = 24)	1.27	11 M

5.3.1. Computational Time

The translation task shows excellent improvement in training speed. As stated in Table 10, the fine-tuning process is 55% faster, and the inference speed increases by 9%. Our architecture's novelty is in reducing the decoder's size to maximize efficiency that primarily targets inference. However, the WMT16 dataset, on average, consists of a 50% lesser word count in target sequences (25 words, compared to CNN/DM's 50). It will reduce the decoder's workload, which might appear to be an unsatisfactory speed-up rate. This result shows that the more extended the sequences are, the more impactful our approach will be on inference speed.

Table 10. Comparing vanilla BART and proposed architecture on the minutes required to fine-tune the model for 1 epoch and performing inference on the full test set (translation task).

Model	Computational Time (Minutes)	
	Fine-Tuning	Inference
BART	247	23
BART + AE (C = 384)	110	21

5.3.2. GPU Memory Usage

As presented in Table 11, with only 10% translation quality loss, the proposed architecture uses 37% and 14% less GPU memory while training and inference, respectively. The translation dataset's short source and target sequences require less GPU memory to process and store the weights. Nevertheless, there is a meaningful decrease in memory usage during the model's life cycle.

Table 11. Comparing vanilla BART and the proposed architecture on the GPU memory (MBs)used to fine-tune the model for 1 epoch and performing inference on the full test set (translation task).

Model	GPU Memory Usage (MB)	
	Fine-Tuning	Inference
BART	25,024	31,680
BART + AE (*C* = 384)	13,120	27,200

5.4. Classification

The BART-base model with a classification head was used for the classification task. Compared to previous sections, the main difference is using a feedforward layer instead of a decoder. There are three experiments presented in Table 12. First, the baseline accuracy for fine-tuning the model. Second, freezing the BART encoder to fine-tune the head component that reduces the accuracy by 6%. Third, the addition of the pre-trained autoencoder with various compression sizes. The 35% compression rate (C = 504) reduces the head size by 55% while showing an 8% reduction in accuracy. The following experiments are performed using a batch size of 64 and 128 for training and inference, respectively.

Table 12. The accuracy results from integrating BART-base with our proposed architecture for classification. The autoencoders are frozen, and "fe" stands for "frozen encoder". The last column indicates the number of parameters in the classification head part of the network.

Models		Accuracy (%)	# Classifier Head Params
BART		86.73	592 K
$BART_{fe}$		80.19	592 K
$BART_{fe}$	+AE (C = 504)	78.12	263 K
	+AE (C = 384)	75.94	148 K
	+AE (C = 120)	70.04	16 K
	+AE (C = 24)	59.43	1 K

The proposed architecture does not make a significant contribution to the classification task. It is better to freeze the encoder and sacrifice fewer accuracy points. It is worth noting that the addition of the autoencoder results in a behavior as seen in previous experiments regarding accuracy drop. However, it does not improve efficiency by a large margin since the classifier head size is minimal, and there is little room for improvement. Also, it should be noted that the autoencoder adds about 1.5 M parameters (depending on compression rate) to the architecture.

5.4.1. Computational Time

The computational time results are displayed in Table 13. There is no noticeable change in the prediction duration. However, it will result in a more efficient training process since we are fine-tuning a small percentage of the network while the rest is frozen. The addition of an autoencoder performs marginally worse.

Table 13. Comparing vanilla BART and the proposed architecture on the minutes it takes to fine-tune the model for 1 epoch and performing inference on the full test set (classification task).

Model	Computational Time (Minutes)	
	Fine-Tuning	Inference
BART	1200	83
$BART_{fe}$	450	83
BART + AE (C = 384)	480	85

5.4.2. GPU Memory Usage

The Table 14 presents the GPU usage results. It follows the same trend as the computational time benchmarks and shows no advantage in using the proposed architecture for the classification task.

Table 14. Comparing vanilla BART and the proposed architecture on the GPU memory (MBs) used to fine-tune the model for 1 epoch and performing inference on the full test set (classification task).

Model	GPU Memory Usage (MB)	
	Fine-Tuning	Inference
BART	27,264	22,831
$BART_{fe}$	3244	22,831
BART + AE (C = 384)	3264	22,966

6. Conclusions

This paper presents a method to shrink the decoder size (number of parameters) in a sequence-to-sequence setting. It has been performed by incorporating an autoencoder as a dimensionality reduction method between the encoder and decoder. The proposed method was validated using a custom summarizer network with various pre-trained encoders. The main idea is that decreasing the size of the encoder's output leads to a dramatic decrease in the decoder size. It results in an optimized network in terms of GPU memory consumption and computational time.

Later, we thoroughly studied the strengths and shortcomings of the proposed architecture on the BART-base model. It is shown that the mentioned architecture on summarization can reduce the decoder's size by 60% (from 96 M to 40 M) while maintaining 95.5% (in terms of R1 score) of the text generation capability. This will significantly impact speeding up both (i) fine-tuning by 45% while using 57% less GPU memory and (ii) the inference process by 54% while utilizing 20% less GPU memory. It will make the large pre-trained models more accessible to anyone without high-end hardware with negligible quality.

The automatic translation experiment follows the same trend as summarization. However, the binary classification task did not improve the model's efficiency while showing the same trade-off between compression rate and accuracy loss. Furthermore, the additional experiments revealed the autoencoder's impressive generalization capability on different datasets.

Our method can be directly used with other approaches, such as distillation, pruning and quantization, to reduce the network size further. One of our future research projects will investigate which combination could lead to the best compromise between the size and the model's accuracy.

Author Contributions: Conceptualization, A.A.F. and R.G.; methodology, A.A.F. and R.G.; software, A.A.F.; validation, A.A.F.; formal analysis, A.A.F. and R.G.; investigation, A.A.F.; resources, R.G.; data curation, A.A.F.; writing—original draft preparation, A.A.F.; writing—review and editing, A.A.F. and R.G.; visualization, A.A.F.; supervision, R.G.; project administration, R.G.; funding acquisition, R.G. All authors have read and agreed to the published version of the manuscript.

Funding: Natural Sciences and Engineering Research Council of Canada (NSERC) grant number 08.1620.00000.814006.

Data Availability Statement: No new data were created or analyzed in this study. Data sharing is not applicable to this article.

Conflicts of Interest: The authors declare no conflict of interest.

Appendix A. Autoencoder Architecture

Table A1 shows the full results of different building blocks and sizes for autoencoder architecture. The linear layer AE outperforms LSTM and CNN in almost all the compression sizes. While the only exception is the LSTM network with the smallest latent

space dimension (32) that narrowly achieved a better accuracy, none of the experiments had an acceptable performance in that representation size. Also, it shows that the CNN architecture resulted in the worst score by a large margin.

Table A1. The MSE loss value of the selected 3 network types (LSTM, linear, CNN) with a different number of layers.

Type	Compression Rate	Number of Layers	MSE Loss
Linear	32	6	0.0930
		4	0.0752
	64	6	0.0766
		8	0.0775
	128	6	0.0637
	256	6	0.0453
	384	6	0.0278
	448	6	0.0239
	512	6	0.0200
LSTM	32	6	0.0905
		2	0.1176
		4	0.0863
	64	6	0.0810
		8	0.0849
		10	0.1043
	128	6	0.0670
	256	6	0.0543
	384	6	0.0462
	448	6	0.0427
	512	6	0.0400
CNN	64	4	0.2666
		6	0.2659
		8	0.2750

Table A2 shows the details of the 6-layer linear architecture comprising three projections in each encoder and decoder (illustrated in Figure 2). The idea is to keep a gentle decrease in size for large latent spaces and to have enough learning capacity with wider networks in smaller compressed sizes.

Table A2. The autoencoder models projections for different compression rates.

First Projection (P1)	Second Projection (P2)	Third Projection/ Compressed Latent Space Size (C)
640	576	512
608	528	448
576	480	384
640	320	256
512	256	128
512	256	64
512	256	32

Appendix B. Full Results of the Validation Experiments

In these results (Tables A3–A5), we used K values equal to 5 and 2 for the beam search and weighted random sampling based on previous experiments. It is also worth noting that the greedy inference method constantly results in better scores, with the weighted random sampling method following closely. The fact that the beam search algorithm leans toward shorter sequences [55] reduces the ROUGE scores since there are fewer matching N-grams in the generated and target summaries. It is partly because of not using any method to force/penalize the token generation process in these experiments (this is not the case in experiments in Section 5.2). It does not mean that the sentence structure/quality is

flawless by using greedy/weighted random sampling, or poor by using beam search. The results simply reflect what the ROUGE score is measuring: an N-gram overlap between the generated and the target sequences.

Table A3. The ROUGE score for using a pre-trained autoencoder on top of pre-trained transformer-based encoders with different compression sizes. Each network was tested using the greedy method. The percentages indicate the relative change in the ROUGE score compared to the baselines.

Models	**Greedy**		
	R-1	**R-2**	**R-L**
Transformer	0.346	0.143	0.312
+AE ($C = 512$)	0.368 (106%)	0.157 (109%)	0.325 (104%)
+AE ($C = 384$)	0.363 (104%)	0.154 (107%)	0.322 (103%)
+AE ($C = 128$)	0.308 (89%)	0.114 (79%)	0.286 (91%)
+AE ($C = 32$)	0.156 (45%)	0.019 (13%)	0.174 (55%)
$BART_{enc}$	0.355	0.142	0.310
+AE ($C = 512$)	0.341 (96%)	0.128 (90%)	0.298 (96%)
+AE ($C = 384$)	0.332 (93%)	0.121 (85%)	0.291 (93%)
+AE ($C = 128$)	0.257 (72%)	0.063 (44%)	0.239 (77%)
+AE ($C = 32$)	0.145 (40%)	0.014 (9%)	0.168 (54%)
BERT	0.349	0.133	0.306
+AE ($C = 512$)	0.339 (97%)	0.123 (92%)	0.298 (97%)
+AE ($C = 384$)	0.332 (95%)	0.119 (89%)	0.294 (96%)
+AE ($C = 128$)	0.278 (79%)	0.074 (55%)	0.256 (83%)
+AE ($C = 32$)	0.168 (48%)	0.021 (15%)	0.187 (61%)
DistilBERT	0.317	0.124	0.283
+AE ($C = 512$)	0.333 (105%)	0.123 (99%)	0.298 (105%)
+AE ($C = 384$)	0.334 (105%)	0.122 (98%)	0.297 (104%)
+AE ($C = 128$)	0.287 (90%)	0.083 (66%)	0.265 (93%)
+AE ($C = 32$)	0.161 (50%)	0.020 (16%)	0.180 (63%)

Table A4. The ROUGE score for using a pre-trained autoencoder on top of pre-trained transformer-based encoders with different compression sizes. Each network was tested using weighted random sampling method. The percentages indicate the relative change in the ROUGE score compared to the baselines.

Models	**Random Sampling**		
	R-1	**R-2**	**R-L**
Transformer	0.344	0.136	0.304
+AE ($C = 512$)	0.363 (105%)	0.147 (108%)	0.315 (103%)
+AE ($C = 384$)	0.360 (104%)	0.146 (107%)	0.314 (103%)
+AE ($C = 128$)	0.315 (91%)	0.110 (80%)	0.280 (92%)
+AE ($C = 32$)	0.184 (53%)	0.024 (17%)	0.185 (60%)
$BART_{enc}$	0.349	0.134	0.301
+AE ($C = 512$)	0.337 (96%)	0.120 (89%)	0.289 (96%)
+AE ($C = 384$)	0.327 (93%)	0.112 (83%)	0.282 (93%)
+AE ($C = 128$)	0.260 (74%)	0.058 (43%)	0.232 (77%)
+AE ($C = 32$)	0.174 (49%)	0.019 (14%)	0.179 (59%)
BERT	0.347	0.124	0.297
+AE ($C = 512$)	0.339 (97%)	0.116 (93%)	0.289 (97%)
+AE ($C = 384$)	0.333 (95%)	0.112 (90%)	0.286 (96%)
+AE ($C = 128$)	0.288 (82%)	0.072 (58%)	0.252 (84%)
+AE ($C = 32$)	0.197 (56%)	0.026 (20%)	0.194 (65%)

Table A4. *Cont.*

Models	Random Sampling		
	R-1	**R-2**	**R-L**
DistilBERT	0.316	0.117	0.275
+AE (C = 512)	0.332 (105%)	0.116 (99%)	0.290 (105%)
+AE (C = 384)	0.334 (105%)	0.115 (98%)	0.288 (104%)
+AE (C = 128)	0.297 (93%)	0.081 (69%)	0.259 (94%)
+AE (C = 32)	0.189 (59%)	0.024 (20%)	0.188 (68%)

Table A5. The ROUGE score for using a pre-trained autoencoder on top of pre-trained transformer-based encoders with different compression sizes. Each network was tested using the beam search method. The percentages indicate the relative change in the ROUGE score compared to the baselines.

Models	Beam Search		
	R-1	**R-2**	**R-L**
Transformer	0.259	0.116	0.261
+AE (C = 512)	0.288 (111%)	0.127 (109%)	0.280 (107%)
+AE (C = 384)	0.278 (107%)	0.123 (106%)	0.274 (104%)
+AE (C = 128)	0.280 (108%)	0.116 (100%)	0.271 (103%)
+AE (C = 32)	0.132 (50%)	0.021 (18%)	0.147 (56%)
$BART_{enc}$	0.304	0.128	0.283
+AE (C = 512)	0.312 (102%)	0.126 (98%)	0.285 (101%)
+AE (C = 384)	0.278 (91%)	0.123 (96%)	0.274 (96%)
+AE (C = 128)	0.245 (80%)	0.071 (55%)	0.234 (82%)
+AE (C = 32)	0.128 (42%)	0.015 (11%)	0.146 (51%)
BERT	0.283	0.117	0.270
+AE (C = 512)	0.291 (102%)	0.117 (100%)	0.275 (101%)
+AE (C = 384)	0.272 (96%)	0.107 (91%)	0.263 (97%)
+AE (C = 128)	0.242 (85%)	0.076 (64%)	0.237 (87%)
+AE (C = 32)	0.140 (49%)	0.020 (17%)	0.153 (56%)
DistilBERT	0.302	0.123	0.280
+AE (C = 512)	0.287 (95%)	0.116 (94%)	0.275 (98%)
+AE (C = 384)	0.282 (93%)	0.114 (92%)	0.270 (96%)
+AE (C = 128)	0.240 (79%)	0.082 (66%)	0.238 (85%)
+AE (C = 32)	0.134 (44%)	0.019 (15%)	0.147 (52%)

Appendix C. BERTScore Results (Validation Experiment)

The following table (Table A6) demonstrates the evaluation results of the proposed method using the BERTScore metric. The mentioned metric is also confirming the paper's discussions on the contextual level.

Appendix D. Extra Examples of Generated Summaries (Validation Experiment)

The samples of generated summaries using the greedy, weighted random sampling, and beam search inference methods are presented in Tables A7–A9. All tables use an autoencoder with a latent space size of 384, which showed promising results in our experiments. The generated summaries' maximum length is set to 60 tokens in all experiments. All the generated summaries are grammatically correct and capture the primary information of the original texts.

Table A6. Comparing the vanilla and proposed models generated the quality of summaries using the BERTScore metric.

Model	Inference Method	Vanilla	+AE (C = 512)	+AE (C = 384)	+AE (C = 128)	+AE (C = 32)
Transformer	Greedy	0.858	0.861	0.860	0.846	0.801
	Random	0.857	0.86	0.859	0.847	0.809
	Beam	0.852	0.858	0.857	0.853	0.805
BART	Greedy	0.867	0.865	0.863	0.845	0.814
	Random	0.869	0.864	0.862	0.845	0.819
	Beam	0.866	0.866	0.865	0.851	0.821
BERT	Greedy	0.858	0.857	0.854	0.854	0.809
	Random	0.857	0.856	0.854	0.843	0.815
	Beam	0.841	0.854	0.854	0.846	0.814
DistilBERT	Greedy	0.798	0.855	0.855	0.842	0.809
	Random	0.836	0.855	0.854	0.844	0.815
	Beam	0.802	0.856	0.855	0.846	0.814

Table A7. Comparing the generated summaries of the vanilla model and the model with a latent space size of 384 using the greedy decoding method.

Model	Vanilla Model-Generated Summary (Greedy)	+AE Model-Generated Summary (Greedy)
Transformer	masked men armed with handguns have robbed three banks in pittsburgh area. they are believed to have had military training and are being described as 'armed and extremely dangerous'. the men are believed to have threatened to kidnapping those at their targets and shoot police. however, the way that the men handle	two men armed with handguns robbed three banks in pittsburgh area so far this year. the unknown men, who are seen on surveillance footage pointing their guns at bank employees' heads, have threatened to kidnapping those at their targets and shoot police. however, the way the men handle their weapons has led
$BART_{enc}$	two men are believed to have had military training and are being described by the fbi as 'armed and extremely dangerous'. the men are seen holding their finger stretched along the barrel of his gun, just off of the trigger, a safety method used by law enforcement. the men, who wear dark	two pennsylvania bank robber are believed to have had military training. they are believed to have been armed and extremely dangerous. the men are believed to have been armed with a pair of masked men armed with handgun. the men are believed to have been from pittsburgh.
BERT	two pennsylvania bank robbers have robbed three banks in the pittsburgh area so far this year. the unknown men, who are seen on surveillance footage, have threatened to kidnapping those at their targets and shoot police. the two men, both 5′ 5′–9′ and april 10, are described as	two pennsylvania bank robbers armed with handguns have been robbed in the pittsburgh area so far this year. they have been seen jumping over the counter as they take their guns at targets and shoot them at police. the two men, who are seen on surveillance footage, have threatened to kidnapping those at their
DistilBERT	two robbers have been seen in a series of recent heisting robberies. the men are believed to have had military training and are being described as 'armed and extremely dangerous'. the men are believed to have been armed and armed. the men are believed to have been armed and extremely dangerous.	two masked men armed with handguns have robbed three banks in the pittsburgh area so far this year. they are believed to have had military training and are being described by fbi as 'armed and extremely dangerous'. the men are believed to have had military training and are being described by the fbi as

Table A8. Comparing the generated summaries of the vanilla model and the model with a latent space size of 384 using the weighted random sampling decoding method.

Model	Vanilla Model-Generated Summary (Weighted Random Sampling)	+AE Model-Generated Summary (Weighted Random Sampling)
Transformer	masked men armed with handguns have robbed three banks in pittsburgh area so far this year. they are believed to be armed and extremely dangerous. they are thought to have been armed with handguns and are thought to be from pittsburgh. the suspects are described as white, 5′ 8′ to 5	the men, who are seen on surveillance footage pointing guns at bank employees' heads, have threatened to kidnapping those at their targets and shoot police. however, the way that the two men handle their weapons has led the fbi to suspect that the thieves are actually former police officers themselves. they are also
$BART_{enc}$	two men have been robbed by the fbi since april 10, according to surveillance footage. they have been seen holding his finger stretched along the barrel of his gun. they have been seen jumping over the counter as they begin their heists. the two robbers have a gun worn during the robberies	two pennsylvania bank robbery suspects have been seen in a string of recent heists. the suspects are believed to have been from pittsburgh. the suspects are believed to be from pittsburgh because of their attitudes.
BERT	the unknown men, who are seen on surveillance footage, have threatened to kidnap those at their targets and shoot police. the two men, both 5′ 5′–9′ and april 10, have also been taken to the bank in pittsburgh, pennsylvania. the fbi believes the two suspects may have	two pennsylvania bank robbers armed as they do a series of recent robberies. they have been described as 'armed and extremely dangerous'. they have been seen on surveillance footage showing the two men. they have been described as 'armed and extremely dangerous' and dangerous.
DistilBERT	the men, who wear dark sweatpants, are believed to be armed and extremely violent. the two men are thought to have been armed and armed. they are believed to be from pittsburgh, pennsylvania, who have been robbed three banks. the men are thought to have been wearing the gun and a gun.	two masked men are thought to have robbed three banks in pittsburgh this year. they are believed to have been armed and extremely dangerous. they have been described as armed and extremely dangerous.

Table A9. Comparing the generated summaries of the vanilla model and the model with a latent space size of 384 using the weighted random sampling decoding method.

Model	Vanilla Model-Generated Summary (Weighted Random Sampling)	+AE Model-Generated Summary (Weighted Random Sampling)
Transformer	masked men armed with handguns have robbed three banks in pittsburgh area so far this year, most recently on april 10	two men armed with handguns robbed three banks in pittsburgh area so far this year, most recently on april 10
$BART_{enc}$	the men, who are seen on surveillance footage pointing their guns at bank employees' heads, have threatened to kidnapping those at their targets and shoot police. the two men are actually former police officers themselves.	two pennsylvania bank thieves are believed to have had military training and are being described by the fbi as 'armed and extremely dangerous'.
BERT	two pennsylvania bank robbers are believed to have had military training and are being described by	two pennsylvania bank robbers armed with handguns have been robbed in the pittsburgh area so far this year, most recently on april 10
DistilBERT	a pair of masked men armed with handguns have robbed three banks in the pittsburgh area so far this year, most recently on april 10. the unknown men, who are seen on surveillance footage pointing their guns at bank employees' heads, have threatened to kidnapping and shoot police.	two masked men armed with handguns have robbed three banks in the pittsburgh area so far this year, most recently on april 10

Appendix E. Extra Results for BERT Model + AE (Validation Experiment)

We conducted a few more experiments on the BERT model to find the optimal latent space size and ROUGE score combination. As shown in Table 4, the intermediate *C* values also follow the same trend as presented in Table A10.

Table A10. The ROUGE score of more experiments on the optimal latent space size options.

Model	Inference Method	R-1	R-2	R-L
BERT + AE (*C* = 448)	Greedy	0.337	0.123	0.297
	Random	0.337	0.115	0.289
	Beam	0.283	0.113	0.270
BERT + AE (*C* = 256)	Greedy	0.323	0.109	0.285
	Random	0.323	0.101	0.277
	Beam	0.272	0.103	0.262
BERT + AE (*C* = 64)	Greedy	0.250	0.048	0.226
	Random	0.234	0.047	0.227
	Greedy	0.195	0.047	0.199

Appendix F. Samples of Generated Summaries (BART-Base Experiment)

The following summaries (Table A11) are generated using the beam searching decoding method with K = 4 and a max length limit of 144 tokens. There is no noticeable quality loss in the summaries generated using the smaller model.

Table A11. Comparing the quality of generated summaries of the BART-base model with the BART + AE with a 384 compression size. The summaries are generated using beam search.

Vanilla BART-Base	+AE (C = 384)
Rifaat al-Assad, 77, was kicked out of Syria 'with nothing' 30 years ago. He went into exile after staging failed coup against brother Hafez al Assad. Activists say his fortune was stolen during his time at heart of Syrian regime. Mr Al-Assad has vehemently denied acquiring assets in France through illegal means. Lawyer says his client's property holdings dated back to 1984–1986.	Rifaat al-Assad, 77, went into exile in Europe after staging a failed coup. He has spent more than 30 years living a life of luxury moving between homes in Paris, London and the southern Spanish city of Marbella. His family's assets, outlined by French customs in May 2014, are valued at around £64 million—much of it held through a web of businesses based in Luxembourg. Al-Assad has vehemently denied acquiring assets in France through illegal means.
Rand Paul, a libertarian-leaning Kentucky senator, launched his presidential bid Tuesday in Louisville, Kentucky. He sparred with TODAY host Savannah Guthrie about his past foreign policy positions. 'Why don't we let me explain instead of talking over me, OK?' he griped. Guthrie obliged, asking him if he had changed his views, but he charged ahead.	Rand Paul, a libertarian-leaning Kentucky senator, sparred with Today host Savannah Guthrie about his past foreign policy positions. Paul, who launched his presidential bid on Tuesday in Louisville, Kentucky, was joined by his wife Kelley Ashby on stage Tuesday as he declared that he would campaign to 'take our country back' 'If they're immediately saying that the agreement doesn't mean what President Obama says, that is a big problem,' Paul said Wednesday.
The search area for missing Malaysia Airlines Flight 370 looks set to double in size. The search will stretch into a new equally vast area, officials from Malaysia, Australia and China say. Families of passengers and crew members still have no answers about what happened to their loved ones.	The search area for missing Malaysia Airlines Flight 370 looks set to double in size. So far, they've covered 60% of the priority search zone without reporting any trace of the airliner. The search of the 60,000-square- kilometer area is expected to be completed in May.
Japanese Prime Minister Shinzo Abe is scheduled to speak Wednesday to a joint meeting of Congress. Julian Zelizer: Abe arrives in Washington at an opportune time to help along the economic centerpiece of the "pivot" Zelizer: The immediate battle in Congress is not over the TPP directly, but something called trade promotion authority.	David Rothkopf: Japanese Prime Minister Shinzo Abe to speak Wednesday to Congress. He says U.S.-Japan relations have been strained by trade promotion authority, but it's not over. RothkopF: Obama administration needs to sell "pivot" or "rebalance" to Americans.

Table A11. *Cont.*

Vanilla BART-Base	+AE (C = 384)
Inverness Caley Thistle defender Josh Meekings has been banned for one match. Meekings was charged over the handball that thwarted a Leigh Griffiths effort. Celtic wrote to the SFA seeking 'an understanding' of why no penalty and red card followed. FIFA vice-president Jim Boyce says the suspension is wrong.	Inverness defender Josh Meekings has been suspended for one match. The defender was charged over the handball that thwarted a Leigh Griffiths effort in their semi-final victory. FIFA vice-president Jim Boyce says the ban should be made if the Scottish FA feel the officials in charge of this game acted improperly and made the wrong decision.

Appendix G. Samples of Generated Translation (BART-Base Experiment)

The following translations (Table A12) are generated using the beam searching decoding method with K = 4 and a max length limit of 128 tokens. The translations generated using the smaller model have no noticeable quality loss.

Table A12. Comparing the quality of generated translations of the BART-base model with the BART+AE with a 384 compression size. The translations are generated using beam search.

Vanilla BART-Base	+AE (C = 384)
s-a ajuns, de fapt, dintr-o reuniune unică de 12 persoane convocată pentru a dezvolta un model de analiză a deciziilor cu mai multe criterii pentru a-şi sintet aliza opiniile cu privire la efectele asociate cu diferite produse cu conţinut de ngl; rezultatele reuniunii au fost rezumate într-un document de cercetare.	rea vine de la o singură reuniune a 12 persoane convocată pentru a dezvolta un model multi-critic de luare a deciziilor (mala da) pentru a-şi sintet aliza opiniile cu privire la riscurile asociate cu diferite produse care conţin fumul de tutun; rezultatele reuniunii au fost rezumate într-o lucrare de cercetare.
ball y a apărat abordarea părţii sale şi a declarat că s-au axat doar pe " contactul " lor atunci când au intrat în conflict.	y a apărat abordarea echipei sale şi a declarat că s-a concentrat doar asupra " contactului " lor atunci când se luptă.
în urmă cu câteva zile, fostul director al oficiului, 41 iza nedelcheva, şi alţi foşti sau actuali angajaţi ai cp ci au fost urmăriţi penal în acest caz, fiind acuzaţi că au plătit pentru serviciile percepute de companiile menţionate, deşi lucrările nu au fost niciodată realizate.	la câteva zile în urmă, fostul şef al biroului, na riza ne zele cu, şi alţi foşti sau actuali angajaţi ai co ci au fost judecaţi în acest caz, fiind acuzaţi de plata serviciilor percepute de companiile menţionate, deşi lucrările nu au fost niciodată efectuate.
unul din doi fumători de-a lungul vieţii moare din cauza dependenţei.	unul din doi fumători de-a lungul vieţii moare din cauza dependenţei.
vom discuta şi vom vedea.	o să vorbim şi să vedem.

References

1. Falaki, A.A.; Gras, R. A Robust Approach to Fine-Tune Pre-Trained Transformer-Based models for Text Summarization through Latent Space Compression. In Proceedings of the 2022 21st IEEE International Conference on Machine Learning and Applications (ICMLA), Nassau, Bahamas, 12–14 December 2022; pp. 160–167. [CrossRef]
2. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, Ł.; Polosukhin, I. Attention is All you Need. In *Advances in Neural Information Processing Systems*; Curran Associates, Inc.: Red Hook, NY, USA, 2017; Volume 30, Available online: https://proceedings.neurips.cc/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf (accessed on 1 April 2023).
3. Radford, A.; Wu, J.; Child, R.; Luan, D.; Amodei, D.; Sutskever, I. Language models are unsupervised multitask learners. *OpenAI Blog* **2019**, *1*, 9.
4. Brown, T.B.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A.; et al. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*; Curran Associates, Inc.: Red Hook, NY, USA, 2020; Volume 33, pp. 1877–1901.
5. OpenAI "GPT-4 Technical Report." ArXiv. 2023. Available online: https://arxiv.org/abs/2303.08774 (accessed on 1 April 2023).
6. Touvron, H.; Lavril, T.; Izacard, G.; Martinet, X.; Lachaux, M.-A.; Lacroix, T.; Rozière, B.; Goyal, N.; Hambro, E.; Azhar, F.; et al. Llama: Open and efficient foundation language models. *arXiv* **2023**, arXiv:230213971.
7. Strubell, E.; Ganesh, A.; McCallum, A. Energy and policy considerations for deep learning in NLP. *arXiv* **2019**, arXiv:190602243.
8. Fedus, W.; Zoph, B.; Shazeer, N. Switch Transformers: Scaling to Trillion Parameter Models with Simple and Efficient Sparsity. *arXiv* **2021**, arXiv:210103961.

9. Lewis, M.; Liu, Y.; Goyal, N.; Ghazvininejad, M.; Mohamed, A.; Levy, O.; Stoyanov, V.; Zettlemoyer, L. 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; Association for Computational Linguistics, Online, 5–10 July 2020; pp. 7871–7880.
10. Zhang, J.; Zhao, Y.; Saleh, M.; Liu, P. Pegasus: Pre-training with extracted gap-sentences for abstractive summarization. In *International Conference on Machine Learning*; Proceedings of Machine Learning Research: Cambridge, MA, USA, 2020; pp. 11328–11339.
11. Liou, C.-Y.; Cheng, W.-C.; Liou, J.-W.; Liou, D.-R. Autoencoder for words. *Neurocomputing* **2014**, *139*, 84–96. [CrossRef]
12. Liu, Y.; Liu, P.; Radev, D.; Neubig, G. BRIO: Bringing Order to Abstractive Summarization. *arXiv* **2022**, arXiv:220316804.
13. Goyal, T.; Rajani, N.F.; Liu, W.; Kryściński, W. Hydrasum: Disentangling stylistic features in text summarization using multi-decoder models. *arXiv* **2021**, arXiv:211004400.
14. Liu, Y.; Gu, J.; Goyal, N.; Li, X.; Edunov, S.; Ghazvininejad, M.; Lewis, M.; Zettlemoyer, L. Multilingual denoising pre-training for neural machine translation. *Trans. Assoc. Comput. Linguist.* **2020**, *8*, 726–742. [CrossRef]
15. Edunov, S.; Ott, M.; Auli, M.; Grangier, D. Understanding back-translation at scale. *arXiv* **2018**, arXiv:180809381.
16. Takase, S.; Kiyono, S. Lessons on parameter sharing across layers in transformers. *arXiv* **2021**, arXiv:210406022.
17. Yang, Z.; Dai, Z.; Yang, Y.; Carbonell, J.; Salakhutdinov, R.R.; Le, Q.V. Xlnet: Generalized autoregressive pretraining for language understanding. *Adv. Neural Inf. Process. Syst.* **2019**, *32*, 5753–5763.
18. Mao, Y.; Wang, Y.; Wu, C.; Zhang, C.; Wang, Y.; Zhang, Q.; Yang, Y.; Tong, Y.; Bai, J. LadaBERT: Lightweight Adaptation of BERT through Hybrid Model Compression. In Proceedings of the 28th International Conference on Computational Linguistics, International Committee on Computational Linguistics, Barcelona, Spain (Online), 8–13 December 2020; pp. 3225–3234.
19. Ding, S.; Shang, J.; Wang, S.; Sun, Y.; Tian, H.; Wu, H.; Wang, H. ERNIE-Doc: A Retrospective Long-Document Modeling Transformer. *arXiv* **2020**, arXiv:2012.15688. [CrossRef]
20. Raffel, C.; Shazeer, N.; Roberts, A.; Lee, K.; Narang, S.; Matena, M.; Zhou, Y.; Li, W.; Liu, P.J. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.* **2020**, *21*, 5485–5551.
21. Gupta, S.; Agrawal, A.; Gopalakrishnan, K.; Narayanan, P. Deep learning with limited numerical precision. In *International Conference on Machine Learning*; Proceedings of Machine Learning Research: Cambridge, MA, USA, 2015; pp. 1737–1746.
22. Micikevicius, P.; Narang, S.; Alben, J.; Diamos, G.; Elsen, E.; Garcia, D.; Ginsburg, B.; Houston, M.; Kuchaiev, O.; Venkatesh, G.; et al. Mixed precision training. *arXiv* **2018**, arXiv:1710.03740.
23. Buciluǎ, C.; Caruana, R.; Niculescu-Mizil, A. Model compression. In Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Philadelphia, PA, USA, 20–23 August 2006; pp. 535–541.
24. Chatterjee, D. Making neural machine reading comprehension faster. *arXiv* **2019**, arXiv:190400796.
25. Turc, I.; Chang, M.-W.; Lee, K.; Toutanova, K. Well-read students learn better: On the importance of pre-training compact models. *arXiv* **2019**, arXiv:190808962.
26. Sanh, V.; Debut, L.; Chaumond, J.; Wolf, T. DistilBERT, a distilled version of BERT: Smaller, faster, cheaper and lighter. *arXiv* **2019**, arXiv:191001108.
27. Shleifer, S.; Rush, A.M. Pre-trained summarization distillation. *arXiv* **2020**, arXiv:201013002.
28. LeCun, Y.; Denker, J.S.; Solla, S.A. Optimal brain damage. *Adv. Neural Inf. Process. Syst.* **1990**, *2*, 598–605.
29. Gordon, M.A.; Duh, K.; Andrews, N. Compressing bert: Studying the effects of weight pruning on transfer learning. *arXiv* **2020**, arXiv:200208307.
30. Michel, P.; Levy, O.; Neubig, G. Are sixteen heads really better than one? *arXiv* **2019**, arXiv:190510650.
31. Sajjad, H.; Dalvi, F.; Durrani, N.; Nakov, P. Poor Man's BERT: Smaller and Faster Transformer Models. *arXiv* **2020**, arXiv:200403844.
32. Frankle, J.; Carbin, M. The lottery ticket hypothesis: Finding sparse, trainable neural networks. *arXiv* **2018**, arXiv:180303635.
33. Prasanna, S.; Rogers, A.; Rumshisky, A. When bert plays the lottery, all tickets are winning. *arXiv* **2020**, arXiv:200500561.
34. Lagunas, F.; Charlaix, E.; Sanh, V.; Rush, A.M. Block pruning for faster transformers. *arXiv* **2021**, arXiv:210904838.
35. Zaheer, M.; Guruganesh, G.; Dubey, K.A.; Ainslie, J.; Alberti, C.; Ontanon, S.; Pham, P.; Ravula, A.; Wang, Q.; Yang, L.; et al. Big Bird: Transformers for Longer Sequences. *Adv. Neural Inf. Process. Syst.* **2020**, *33*, 17283–17297.
36. Beltagy, I.; Peters, M.E.; Cohan, A. Longformer: The long-document transformer. *arXiv* **2020**, arXiv:200405150.
37. Wang, S.; Li, B.Z.; Khabsa, M.; Fang, H.; Ma, H. Linformer: Self-attention with linear complexity. *arXiv* **2020**, arXiv:200604768.
38. Kitaev, N.; Kaiser, Ł.; Levskaya, A. Reformer: The efficient transformer. *arXiv* **2020**, arXiv:200104451.
39. Hou, L.; Huang, Z.; Shang, L.; Jiang, X.; Chen, X.; Liu, Q. Dynabert: Dynamic bert with adaptive width and depth. *arXiv* **2020**, arXiv:200404037.
40. Sun, Z.; Yu, H.; Song, X.; Liu, R.; Yang, Y.; Zhou, D. Mobilebert: A compact task-agnostic bert for resource-limited devices. *arXiv* **2020**, arXiv:200402984.
41. Tambe, T.; Hooper, C.; Pentecost, L.; Jia, T.; Yang, E.-Y.; Donato, M.; Sanh, V.; Whatmough, P.; Rush, A.M.; Brooks, D.; et al. EdgeBERT: Sentence-Level Energy Optimizations for Latency-Aware Multi-Task NLP Inference. *arXiv* **2020**, arXiv:201114203.
42. Hochreiter, S.; Schmidhuber, J. Long short-term memory. *Neural Comput.* **1997**, *9*, 1735–1780. [CrossRef] [PubMed]
43. Lecun, Y.; Bottou, L.; Bengio, Y.; Haffner, P. Gradient-based learning applied to document recognition. *Proc. IEEE* **1998**, *86*, 2278–2324. [CrossRef]

44. Ross, D.A.; Lim, J.; Lin, R.-S.; Yang, M.-H. Incremental learning for robust visual tracking. *Int. J. Comput. Vis.* **2008**, *77*, 125–141. [CrossRef]
45. Lin, C.-Y. ROUGE: A Package for Automatic Evaluation of Summaries. In Proceedings of the Text Summarization Branches Out, Association for Computational Linguistics. Barcelona, Spain, 10–17 July 2004; pp. 74–81. Available online: https://aclanthology.org/W04-1013 (accessed on 1 April 2023).
46. Papineni, K.; Roukos, S.; Ward, T.; Zhu, W.-J. Bleu: A Method for Automatic Evaluation of Machine Translation. In Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics, Association for Computational Linguistics. Philadelphia, PA, USA, 6–12 July 2002; pp. 311–318. [CrossRef]
47. Smith, L.N.; Topin, N. Super-Convergence: Very fast training of neural networks using large learning rates. In *Artificial Intelligence and Machine Learning for Multi-Domain Operations Applications*; SPIE: Washington, DC, USA, 2019; Volume 11006, p. 1100612.
48. Kingma, D.P.; Ba, J. Adam: A method for stochastic optimization. *arXiv* **2014**, arXiv:14126980.
49. Pereyra, G.; Tucker, G.; Chorowski, J.; Kaiser, Ł.; Hinton, G. Regularizing neural networks by penalizing confident output distributions. *arXiv* **2017**, arXiv:170106548.
50. Hermann, K.M.; Kocisky, T.; Grefenstette, E.; Espeholt, L.; Kay, W.; Suleyman, M.; Blunsom, P. Teaching machines to read and comprehend. *Adv. Neural Inf. Process. Syst.* **2015**, *28*, 1693–1701.
51. Grusky, M.; Naaman, M.; Artzi, Y. Newsroom: A dataset of 1.3 million summaries with diverse extractive strategies. *arXiv* **2018**, arXiv:180411283.
52. Bojar, O.; Chatterjee, R.; Federmann, C.; Graham, Y.; Haddow, B.; Huck, M.; Jimeno Yepes, A.; Koehn, P.; Logacheva, V.; Monz, C.; et al. Findings of the 2016 Conference on Machine Translation. In *Proceedings of the First Conference on Machine Translation: Volume 2, Shared Task Papers*; Association for Computational Linguistics: Berlin, Germany, 2016; pp. 131–198.
53. Go, A.; Bhayani, R.; Huang, L. Twitter sentiment classification using distant supervision. *CS224N Proj. Rep. Stanf.* **2009**, *1*, 2009.
54. Zhang, T.; Kishore, V.; Wu, F.; Weinberger, K.Q.; Artzi, Y. Bertscore: Evaluating text generation with bert. *arXiv* **2019**, arXiv:190409675.
55. Wu, Y.; Schuster, M.; Chen, Z.; Le, Q.V.; Norouzi, M.; Macherey, W.; Krikun, M.; Cao, Y.; Gao, Q.; Macherey, K.; et al. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv* **2016**, arXiv:160908144.