RE: A Study for Restorable Embeddings

Anonymous ACL submission

Abstract

As the number of model parameters increased, large language models achieved linguistic fluency and exhibited high performance in various natural language tasks without gradient updates because the models could retain more knowledge. However, the large model size makes difficult to apply the model to a task requiring domain knowledge not included in the training corpus, due to the fact that knowledge stored in model parameters is not controllable during generation and model parameter updates are costly. To tackle the problem, we suggest separating the language model and knowledge, and divide the end-to-end language model into three parts: 1) encoding knowledge, 2) processing the encoded knowledge, and 3) restoring the processed knowledge embedding to natural language. In this paper, we propose a model for learning restorable embeddings as a first step toward the study to separate the language model and knowledge. The experimental results shows that the proposed model can restore most knowledge in 1-2 sentences by encoding knowledge in sentence-level embeddings and then restoring the embeddings back to the original sentence. We also verify that the embeddings generated through our method significantly improves performance in the passage retrieval task.

1 Introduction

Recently decoder-based language models (Radford et al., 2019; Wang and Komatsuzaki, 2021) and encoder-decoder-based language models (Raffel et al., 2020; Zhang et al., 2020; Lewis et al., 2020) have become linguistically fluent by implicitly storing general knowledge in model parameters and using the stored knowledge during generation. In particular, the number of decoder-based model parameters has increased to store knowledge as much as possible from a large corpus, and resulted in high performance in zero-shot and few-shot settings. However, the number of model parameters has reached 175B (Brown et al., 2020) and 530B (Narayanan et al., 2021).

The cost of updating all parameters through transfer learning became extremely costly due to the large size of language models. Therefore, it is computationally feasible only when updating head layers, whose input are contextualized representations, or manipulating conditional context input without gradient updates. In case domain-specific knowledge is required, it must be provided through conditional context because the amount of the knowledge in model parameters is likely to be small. As more domain knowledge are needed, the length of the conditional context become longer so that the computation cost increases sharply due to Transformer (Vaswani et al., 2017) structure's quadratic memory complexity with respect to the length of the input sequence. Although several sparse attention studies (Beltagy et al., 2020; Zaheer et al., 2020; Roy et al., 2021) have been conducted to address this problem and the length that can be computed in the same memory size has increased about 8 to 10 times, the length limitation of the conditional context remains.

Large language models have another limitation called the hallucination problem (Maynez et al., 2020; Shuster et al., 2021; Roller et al., 2020), which produces a contradiction or a plausible untruth in the generated text. The problem is caused because knowledge are mixed and stored in internal parameters, and it is unclear which knowledge is chosen for text generation. As a way to tackle this problem, we isolate the knowledge in internal parameters to an external permanent memory, and refer to the isolated knowledge whenever needed. To store knowledge in an external memory, an embedding presenting a certain unit of knowledge, which minimizes information loss, must be devised. The embedding should be applicable to natural language processing, and the embedding generated from the processing should be convertible into natural language that humans can understand. If the embedding is restorable to the original text sequence, this approach also improves memory efficiency because the original text does not have to be stored together with the embedding. Otherwise, pairs of embeddings and original texts must be stored in order to extract the correct answer from the document after finding a document containing an answer in tasks such as open-domain question answering.

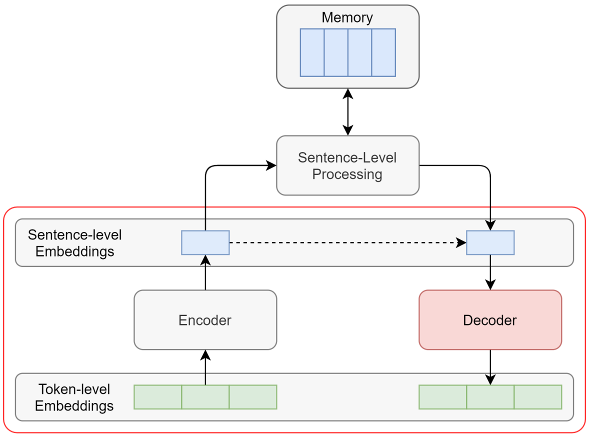

Figure 1: Conceptual diagram of sequence-to-sequence natural language processing using a sentence-level embedding

The framework to separate the language model and knowledge is shown in Figure 1, and illustrates the unit knowledge-based natural language processing which are divided into three stages: (1) creating an embedding vector for sentence-level knowledge to minimize information loss and express its proper meaning; (2) processing a natural language task using the generated embedding and knowledge embedding stored in memory, and expressing the result as embedding; (3) converting the resulting embedding into natural language that humans can understand. If this sentence-level knowledge unit is applied to natural language processing, a larger amount of context can be viewed with the same size of memory. Besides there is no need to look up a large amount of context because context can be converted into sentence-level knowledge embeddings, stored in memory, and processed from memory.

For the framework of Figure 1 to be possible, research on creating embeddings and restoring embeddings back to the original text must be preceded. In this paper, as shown in the red box in Figure 1, we therefore conduct a study to express the token-level embedding sequence as one embedding and to restore the expressed embedding to the original text. If the objective of the model is set to restore the embedding to the original text, the embedding might not suitable in various language tasks because the embedding mainly expresses the lexical information of the original text sequence. Thus, training the model to improve the restoration performance and to maintain or improve the performance for downstream tasks is necessary.

For reconstructable embeddings, (1) we propose a new layer structure to enhance performance of the restoration from the embedding vector to the original text sequence. In addition, (2) we confirm that the generated embedding from the proposed model maintains performance in various downstream tasks and improves performance considerably in passage retrieval where small information loss shows advantageous. Finally, (3) we analyze the length at which the occurrence of hallucination is minimized, according to the length of the original text sequence, when embedding is made and the original text is restored.

2 Related Work

Research on making good sentences and passage embeddings has been studied in various fields such as sentence embedding and passage retrieval. In particular, the sentence embedding study lowered the computational complexity for scoring and classifying between sentence pairs after BERT (Devlin et al., 2019) was introduced. In addition, many studies have been conducted in the fields of long document summarization and document classification as one of the methods to alleviate large memory consumption in long document processing.

2.1 Sentence Embedding

Sentence embedding has been studied for a long time, and various methods such as Skip-thought (Kiros et al., 2015), InferSent (Conneau et al., 2017), and Universal Sentence Encoder (Cer et al., 2018) have been proposed and studied. To alleviate the need to compute all combinations in the classification and similarity scoring task of sentence-pair in BERT, sentence-BERT (Reimers and Gurevych, 2019) proposed classification and similarity scoring methods using sentence embedding. In sentence BERT, a model was trained using the semantic textual similarity (STS) dataset to make good semantic embeddings, and it showed high performance and computational efficiency in various sentence classification and regression tasks.

2.2 Passage Retrieval

Passage Retrieval Task is a task that retrievals passages related to a query in a large number of passages. In Open-domain Question Answering such as Natural Question and TriviaQA, and document augmented conversational models such as WizInt, relevant passages must be searched from large-scale data such as Wikipedia and Common Crawl. Because the number of passages to be ranked is on a million scale, measuring the correlation with all documents for every query requires many calculations. In most methods, queries and passages are thus expressed as embedding vectors and the correlation is measured using metrics such as cosine similarity or inner product between embedding vectors. Recently, several methods (Karpukhin et al., 2020; Xiong et al., 2021; Zhang et al., 2021) for encoding queries and passages using language model encoders have been studied.

2.3 Long-Document Summarization

In long document summarization, the length of the sequence to be summarized is too long, so it is difficult to use Transformer with quadratic memory complexity for the length of the input sequence. Therefore, studies are being conducted in two main directions. One is a study of lowering memory complexity through sparse attention (Wang et al., 2020; Kitaev et al., 2020; Tay et al., 2020; Huang et al., 2021), and the other is a study of making a sentence or paragraph into an embedding vector and then generating a summary using a hierarchical transformer with these embedding vectors (Rohde et al., 2021; Zhang et al., 2019; Liu and Lapata, 2019; Wu et al., 2021). In the case of a method using a hierarchical transformer, a summary is generated end-to-end using an encoder-decoder structure, but research on restoring this embedding vector to a natural language is not in progress.

3 Model Architectures

In this section, we describe the model used in the experiment and the proposed model. The following expressions are used to maintain the consistency of annotations throughout the description.

- $\mathbf{x} = \{x_1, \cdots, x_T\}$: The token sequence to be expressed as an embedding vector
- $\mathbf{y} = \{y_1, \cdots, y_M\}$, $\mathbf{z} = \{z_1, \cdots, z_N\}$: A token sequence to be input to encoder and decoder respectively
- d_{model}: The dimensionality of encoder and decoder
- d_{repr}: the dimensionality of representation vector
- $e(y_i)$: The embedding vector of ith token y_i
- $h(y_i)$: contextualized embedding of y_i produced by encoder
- $\mathbf{e}_{repr}$: The embedding vector of $\mathbf{x}$ generated using encoder

3.1 Passage Encoder

Conventional methods for generating embeddings of text sequences include (a) using the embedding vector of the [CLS] token and (b) using the vector obtained through mean pooling. In case of (a), the [CLS] token and text sequence are concatenated then input to the encoder, and the contextualized embedding value of the [CLS] token position is projected using a linear layer to create an embedding vector. Therefore, the embedding vector $\mathbf{e}_{repr}$ of $\mathbf{x}$ is defined as Eq. 1.

$$\begin{aligned} \mathbf{e}_{repr} &= \mathbf{W}h(y_1) \\ &\text{where } \mathbf{y} = \{[CLS], x_1, \cdots, x_T\} \end{aligned} \tag{1}$$

The projection matrix $\mathbf{W}$ is a learnable variable, and it satisfies $\mathbf{W} \in \mathbb{R}^{d_{model} \times d_{repr}}$. In case of (b), the embedding vector is obtained by inputting the text sequence to the encoder and projecting the vector obtained by mean pooling all contextualized embedding values into a linear layer. Therefore, in case of mean pooling, the embedding vector $\mathbf{e}_{repr}$ of $\mathbf{x}$ is defined as Eq. 2.

$$\mathbf{e}_{repr} = \mathbf{W}(\sum_{i=1}^{T}(h(x_i)/\sqrt{T})) \tag{2}$$

3.2 Passage Decoder

There are two vanilla methods to restore the embedding vector $\mathbf{e}_{repr}$ to the original $\mathbf{x}$ as shown in Figure 2. In Figure 2, (a) uses a decoder structure without cross attention block like GPT. The $\mathbf{e}_{repr}$ and the original text sequence $\mathbf{x}$ is to concatenate and then input it to the decoder, and trained to generate the original sentence from the output. (b) inputs $\mathbf{e}_{repr}$ as the key/value of the cross attention block in the decoder structure, and concatenates [BOS] token and $\mathbf{x}$ as the decoder input, and train the model to generate original sentences as output.

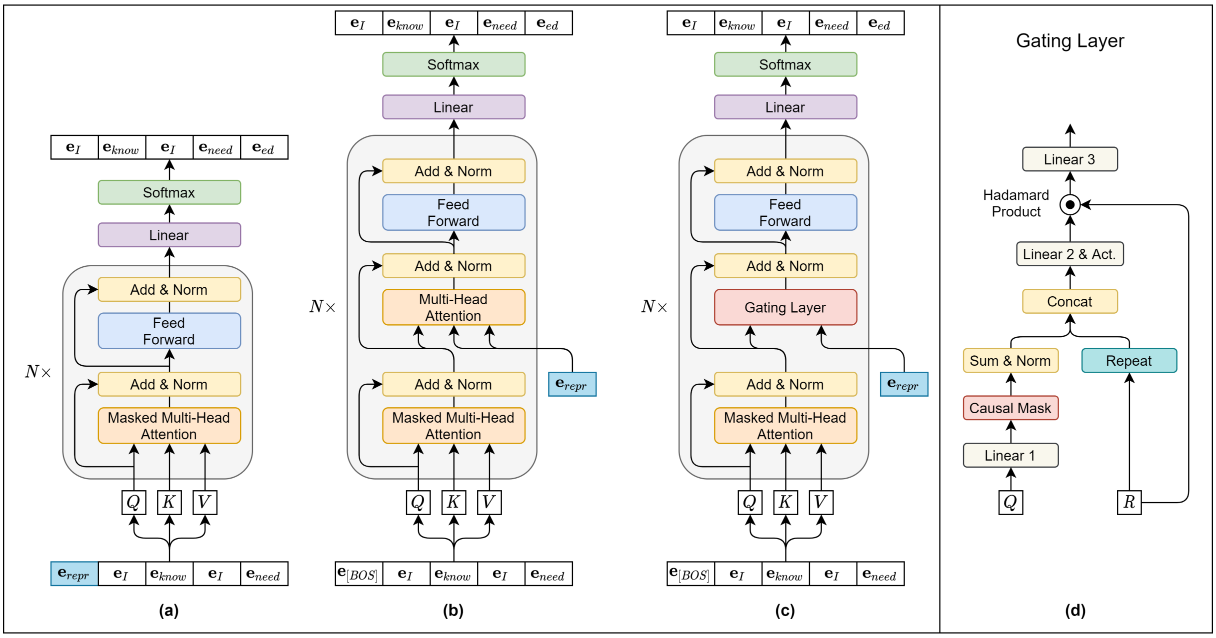

Figure 2: Decoder structures for restoring the embedding vector to the original text. (a) A decoder using embedding vector as input. (b) A decoder using embedding vector as key/value of cross attention layer. (c) The decoder structure using the proposed gating layer instead of the cross attention layer. (d) The structure of the proposed gating layer.

Therefore, in the case of (a), the input sequence is $e(\mathbf{z})$ $\{\mathbf{e}_{repr}, e(x_1), \cdots, e(x_L)\}$, the target sequence is $\{e(x_1), \cdots, e(x_L), e([EOS])\}$. In case of (b), the input sequence is $e(\mathbf{z})$ is $\{e([BOS]), e(x_1), \cdots, e(x_L)\}$, target sequence is $\{e(x_1), \cdots, e(x_L), e([EOS])\}$, and the $\mathbf{e}_{repr}$ is input as key/value of cross attention layer . (Hereafter, in Figure 2, (a) is called an input decoder, and (b) is called a cross decoder.)

However, cross attention calculates attention over sequence dimension and performs sum, so when one embedding is entered as key/value, the query vector and the scalar value, which is the inner product of the embedding vector and the query vector, are multiplied, and this vector is added to the query vector. Therefore, the embedding vector does not reflect only the elements that are highly related to the current query vector, but multiplies and adds all elements of the embedding vector as much as the similarity between the embedding vector and the current query vector. That is, when the sequence of the query vector input to the cross attention layer is $\mathbf{q}_{1:N}$, and the ith query vector is $\mathbf{q}_i$, The query vector $\hat{\mathbf{q}}_i$ updated by cross attention is Eq. 3.

$$\begin{aligned}\hat{\mathbf{q}}_i =& \mathbf{q}_i + c \cdot \mathbf{e}_{repr} \\ & \text{where} \quad c = \mathbf{q}_i \cdot \mathbf{e}_{repr} \\ & \text{s.t.} \quad d_{model} = d_{repr}\end{aligned} \tag{3}$$

As shown in Eq.3, when cross attention is used, a vector multiplied by a scalar to $\mathbf{e}_{repr}$ is added to the query vector. Also, since it is $\mathbf{q}_i \in \mathbb{R}^{d_{model}}$ and $\mathbf{e}_{repr} \in \mathbb{R}^{d_{repr}}$, d_{model} and d_{repr} must be the same in order for inner product between two vectors to be possible. In this paper, we only deal with the case where $d_{model} = d_{repr}$, but it may be necessary to increase the size of d_{repr} to include more information in $\mathbf{e}_{repr}$. This constraint can be a disadvantage in creating embeddings with low information loss. Therefore, we propose a gating layer that can decode even if d_{repr} and d_{repr} are different and extracts only the elements related to the current query vector from the embedding vector.

3.3 Gating Layer

Figure 2 (c) shows the use of the gating layer instead of the cross attention layer, and (d) shows the structure of the gating layer. As the gating layer, query and $\mathbf{e}_{repr}$ are input. When the i-th query vector input to the gating layer is $\mathbf{q}_i \in \mathbb{R}^{d_{model}}$, $\mathbf{q}_i$ is projected to d_{repr} through the projection matrix $\mathbf{W}_1 \in \mathbb{R}^{d_{model} \times d_{repr}}$ and becomes C. $\tilde{\mathbf{q}}_i$ is added to the j-th vectors smaller than i through causal masking and sum operation, and then divided by i, and becomes a normalized vector $\bar{\mathbf{q}}_i$. If $\bar{\mathbf{q}}_i$ is expressed as an expression for $\tilde{\mathbf{q}}_j$, it is the same as Eq. 4.

$$\bar{\mathbf{q}}_i = \sum_{j=1}^{i} \tilde{\mathbf{q}}_j / \sqrt{i} \tag{4}$$

Finally, each $\bar{\mathbf{q}}_i$ is concatenated with $\mathbf{e}_{repr}$, and a vector with $\mathbb{R}^{2d_{repr}}$ dimension is projected to d_{repr} through $\mathbf{W}_2 \in \mathbb{R}^{2d_{repr} \times d_{repr}}$ and then activated through activation function. The activated i-th query vector is gated through the hadamard product with $\mathbf{e}_{repr}$ and finally projected to d_{model} through $\mathbf{W}_3 \in \mathbb{R}^{d_{repr} \times d_{model}}$ to become $\ddot{\mathbf{q}}_i$. If $\ddot{\mathbf{q}}_i$ is expressed as an expression for $\bar{\mathbf{q}}_i$, it becomes Eq. 5.

$$\begin{aligned} \ddot{\mathbf{q}}_i =& (\mathrm{Act}(\dot{\mathbf{q}}_i \mathbf{W}_2) \odot \mathbf{e}_{repr})\mathbf{W}_3 \\ \text{where} \quad & \dot{\mathbf{q}}_i = \mathrm{Concat}(\bar{\mathbf{q}}_i; \mathbf{e}_{repr}) \end{aligned} \tag{5}$$

As shown in (c) of Figure 2, $\ddot{\mathbf{q}}_i$ is added to $\mathbf{q}_i$ and then normalized by layer normalization. Therefore, $\mathbf{e}_{repr}$ gated by the hadamard product is added to $\mathbf{q}_i$. When the structure of Figure 2 (c) including the gating layer is called a gating decoder, the input and target sequence of the gating decoder are the same as that of the cross decoder.

The learning objective of the input, cross, and gating decoder is Eq. 6, which is an auto regressive objective.

$$\max_{\theta} \log p_\theta(\mathbf{x}) = \sum_{t=1}^{T} \log p_\theta(x_t | \mathbf{x}_{<t}, enc_{\hat{\theta}}(\mathbf{x})) \tag{6}$$

$enc_{\hat{\theta}}$ denotes an encoder function parameterized by $\hat{\theta}$, and p_θ denotes the entire encoder-decoder function parameterized by θ. The relationship between θ and $\hat{\theta}$ is $\hat{\theta} \subset \theta$.

4 Experiments

In this section, the embedding of the text sequence created using the proposed model can be restored to the original text, and at the same time, it is shown that the performance is improved in the downstream task using embedding compared to when not used. This shows that the proposed model does not sacrifice downstream performance for recovery performance. The restoration performance of the original text sequence is quantitatively evaluated through Perplexity (PPL), Rouge-1 (R-1), Rouge-2 (R-2), and Rouge-L (R-L) scores. Then, we proceed with qualitative performance evaluation by looking at the actual recovered text. Performance in downstream task using embedding was measured as passage retrieval performance using Natural Question (Kwiatkowski et al., 2019), one of the open domain QA datasets.

4.1 Experimental Settings for Text Restoration

C4 RealNewsLike (Raffel et al., 2020; Zellers et al., 2019) introduced in T5 (Raffel et al., 2020) was used as a raw corpus for text restoration. C4 RealNewsLike is a dataset that applies the preprocessing used in C4 to Common Crawl[1] used in FakeNews (Zellers et al., 2019), and consists of 13 millions samples of train split and 13,863 samples of validation split. The preprocessing used in C4 includes bad word filtering and duplicate removal.

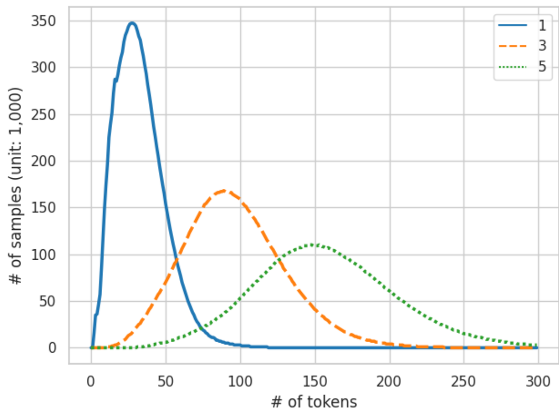

Figure 3: 1, 3, and 5 sentences were used to examine the restoration performance according to the length of the text sequence, and the figure shows the token length distribution for each number of sentences.

In order to examine the restoration performance and the performance in downstream tasks according to the length of the text sequence, the text data was separated into sentence units using NLTK's sentence tokenizer (Bird and Loper, 2004). A dataset was separately constructed according to the number of sentences (1, 3, 5), and Figure 3 shows the token length according to the number of sentences. The average token length according to the number of sentences is 33, 96, and 156 for 1, 3, and 5 sentences, respectively.

The training was conducted for 1 epoch using the train split, and the restoration performance was measured using the validation split. For the model size, a small configuration of T5 was used, and training was carried out after initializing with the pre-trained weights of T5. In order to examine the difference in the restoration performance and the performance difference in the downstream task between whether the pre-trained weights transferred from T5 were frozen or not, both the case of freezing and the case of updating the weights transferred

[1] http://commoncrawl.org/

from T5 were tested. In addition, since there is only the last projection matrix of the encoder as a variable that can be learned to make a restorable embedding in the case of freezing layers, we also measured the restoration performance when 3 Transformer layers are added. The parameters of the 3 Transformer layers were randomly initialized. Therefore, as shown in Table 1, we experimented with 4 configurations for each encoder and decoder variation.

option	(a)	(b)	(c)	(d)
freeze pre-trained weights	N	Y	N	Y
# additional layers w/ random init.	0	0	3	3

Table 1: experiment configuration. There are 4 configurations depending on the combination of whether to add randomly initialized layers after 6 pre-trained layers in the encoder part and whether to update parameters by freezing the pre-trained layers.

Adam optimizer was used as the optimizer, and learning rate scheduling was performed using linear scheduling. d_{model} and d_{repr} were set to 512 in all experiments. Also, Gated ReLU (Dauphin et al., 2017) was used for the activation function in the gating layer. Detailed hyperparameters for model and optimizer can be found in Appendix A.

4.2 Single Sentence Restoration Performance

In Table 2, when the embedding vector is created using the [CLS] token, the restoration performance of the original text is low in all configurations from (a)-(d). Considering that it does not restore well even when three randomly initialized layers are added, global attention is effective in making token-level contextualized embeddings, but there seems to be a limit to making sentence-level embeddings.

Conversely, when embeddings were created using mean pooling, restoration performance was higher in all configurations than when embeddings were created using [CLS] token. Unlike the [CLS] token, since all tokens are used directly to generate embeddings, information loss is low and high restoration performance appears to be achieved. Comparing the restoration performance according to decoders in mean pooling, all experimental configurations and all performance metrics improved in the order of input, cross, and gating methods. That is, the proposed model showed higher restoration performance than the input and cross decoder in all cases.

Comparing the restoration performance according to the experimental configuration when generating embeddings by the mean pooling, the case of freezing pre-trained model weights in both cases with three additional layers and without additional layers, performed lower than those without freezing. This seems to be due to the difference in the number of parameters that can be updated. In case of (b), compared to (a), it shows significantly lower performance. In the case of (a), 6 layers can be updated, but in the case of (b), only the last projection layer can be updated. The large difference in the number of parameters that can be updated seems to be the main cause.

4.3 Performance according to the number of sentences

Table 3 shows the restoration performance when using a cross decoder and a gating decoder for each sentence length. In all cases, as the length of the sentence increases, the recovery performance decreases, which indirectly shows the amount of information that can be contained in a 512-dimensional embedding vector. As the length of the sentence increases, the cross decoder tends to have a relatively sharp decrease in restoration performance than the gating decoder. More restoration performance depending on the text sequence length, experimental configuration, and decoder type can be found in Appendix C.

4.4 Passage Retrieval Performance

The passage retrieval performance was measured to examine the performance in the downstream task using the embedding generated by the proposed model. As in Dense Passage Retrieval (DPR), we used a biencoder that learns two encoders: a query encoder and a passage encoder. The model was trained with in-batch training (Karpukhin et al., 2020) using the positive passages of other samples in the batch as negative passages. Detailed hyper parameters used for training are described in Appendix B. Natural question data and Wikipedia passages data used in DPR were used, so as in DPR, among the 21,015,324 passages, the performance (Recall) of whether passages containing the correct answer to the question exist in the top K passages returned by the model was measured, and the results are shown in Table 4.

First, comparing the performance from the case where there is no additional layer, the case where the sentence restoration was learned performed much higher than the case where the transfer learn-

decoder	classifcation token PPL	R-1	R-2	R-L	mean pooling PPL	R-1	R-2	R-L
(a) 6 layers from pre-trained model + 0 additional layers								
input	6.178	9.87	0.79	8.09	1.16	93.37	82.93	89.72
cross	6.10	7.09	0.19	6.24	1.10	95.14	87.80	92.76
gating	6.04	11.21	0.55	8.21	1.04	97.76	94.63	96.94
(b) 6 layers from pre-trained model (freeze) + 0 additional layers								
input	1.79	13.33	0.75	9.53	2.24	65.99	34.45	50.96
cross	6.22	12.29	0.78	9.30	2.04	67.97	37.85	54.00
gating	6.16	11.13	0.29	8.47	1.93	70.54	40.83	56.81
(c) 6 layers from pre-trained model + 3 additional layers (random initialization)								
input	6.18	13.32	0.75	9.53	1.15	92.63	83.34	89.63
cross	6.10	9.95	0.21	8.31	1.12	94.13	86.26	91.62
gating	6.04	10.81	0.56	8.07	1.03	**98.32**	**96.30**	**97.91**
(d) 6 layers from pre-trained model (freeze) + 3 additional layers (random initialization)								
input	6.30	11.86	0.77	8.84	1.34	84.77	69.68	81.12
cross	6.22	11.21	0.55	8.21	1.29	87.18	73.07	83.79
gating	6.16	9.88	0.58	7.57	1.09	**95.95**	**91.07**	**95.04**

Table 2: The restoration performance of a single sentence according to the experimental configuration, the method used to create the embedding vector, and the decoder type

type	# sents	PPL	R-1	R-2	R-L
decoder - cross					
(c)	1	1.12	94.13	86.26	91.62
	3	1.89	63.08	29.25	46.87
	5	2.80	52.35	15.09	31.28
(d)	1	1.29	87.18	73.07	83.79
	3	2.48	59.00	24.39	44.09
	5	3.50	51.30	14.58	31.00
decoder - gating					
(c)	1	1.03	**98.32**	**96.30**	**97.91**
	3	1.37	**72.11**	**50.45**	**64.16**
	5	2.08	**52.82**	**18.91**	**36.77**
(d)	1	1.09	**95.95**	**91.07**	**95.04**
	3	1.75	**67.14**	**39.97**	**58.43**
	5	2.76	**52.38**	**17.92**	**36.83**

Table 3: Restoration performance of cross decoder and gating decoder according to the number of original sentences (mean pooling was used to generate embeddings)

	# sentences	R@20	R@100
0 additional layers			
T5-small		49.58	67.12
(a)	1	**64.33**	**78.34**
	3	63.09	78.34
	5	63.09	77.88
(b)	1	63.61	78.39
	3	62.56	77.71
	5	62.18	77.67
3 additional layers			
T5-small + 3 layers(random init.)		55.73	72.37
(c)	1	64.07	78.05
	3	63.13	77.82
	5	63.61	78.30
(d)	1	**70.30**	**83.32**
	3	68.70	82.29
	5	68.46	82.13

Table 4: Passage retrieval performance in natural questions according to experimental configuration and sentence length

ing was performed from the T5 small. In addition, even when three random initialized layers were added, the case of learning sentence restoration showed higher performance. The reason why the model that learned sentence restoration showed high performance improvement in passage retrieval seems to be because it was trained to make embeddings with minimal information loss in the passage.

When learning sentence restoration, the frozen case had lower performance, but there was no significant difference when comparing the case where the pre-trained weight part was frozen and the case where it was not frozen. In the case of using three additional layers, the frozen case showed a high performance improvement in the passage retrieval task. It seems that, if the pre-trained weight part is not frozen, the representation that affects passage retrieval performance is damaged during the learning process of the sentence restoration. Therefore, the restoration performance was high in the case of not freezing pre-trained weights, but the performance in passage retrieval was high in the case of freezing. Therefore, when learning a sentence restoration, it is necessary to learn along with a language modeling objective such as masked language modeling or next token prediction, or learn the restoration while maintaining the weight of the already learned language model as in this paper.

4.5 Analysis of the restored text according to the number of sentences

In the case of one sentence, it was completely restored, and almost all samples as well as the samples in Table 5 were restored without loss of information. In the case of 3 sentences, the first sentence was completely restored, but the 2nd and 3rd sen-

		gating decoder
origin	1	Was it a surprise to you that you were given the arts and culture position?
	2	No, there is no surprise when you are a cadre of the ANC because you are deployed anywhere.
	3	You are given a five-year contract to do a portfolio and when you are finished, you wait for another one.
	4	At no stage do you have a say.
	5	What qualities do you bring to the position?
		1 sentence
restored	1	Was it a surprise to you that you were given the arts and culture position?
		3 sentences
restored	1	Was it a surprise to you that you were given the arts and culture position?
	2	No, there is no surprise when you are a cadre of the ANC because you are deployed overseas.
	3	You are given a five-year contract to do a portfolio and when you (are) finish, you are waiting for another.
		5 sentences
restored	1	Was it a surprise to you that you were given the arts and culture culture?
	2	No, there is no surprise when you are a candidate of the ANC because you are deployed anywhere.
	3	You are given a four-year contract to do a portfolio and when you (are) finish(ed), you are no longer looking for one.
	4	At one stage did you have a capabilities?
	5	What does the message bring to you?
		cross decoder
origin	1	Two bedrooms home on a corner lot.
	2	Two car detached garage.
	3	Nice covered front porch.
	4	Seller will not complete any repairs to the subject property, either lender or buyer requested.
	5	The property is sold in AS IS condition.
		5 sentences
restored	1	Two car garage on a corner lot.
	2	Two covered covered porch.
	3	Sony front porch.
	4	Nice covered garage will not return any repairs to the seller, either buyer or seller.
	5	The property is listed in ASOLD condition.

Table 5: Samples in which embedding is restored to origin text according to the length of the input text. Blue text means a part different from the original text, and red text means a part omitted from the original text.

tences omit a part or have different parts with origin text. In particular, the frequency of restoring different from the original in the 3rd sentence was higher than in the 2nd sentence.

In the case of 5 sentences, the 4th and 5th sentences were generated using plausible words except for some keywords. That is, it can be confirmed that the hallucination problem appears due to the loss of information. Comparing the results of encoding 5 sentences of text and restoring it with a cross decoder, it can be confirmed that the information of the original sentences is mixed. Therefore, in the sentence vector dimension and model size used in this experiment, to prevent hallucination problem and minimize information loss, it is appropriate to convert only 1 to 2 sentences into embedding.

5 Conclusion

In this paper, we conducted a study to create restorable embeddings of text sequences. In addition, in order to improve the restoration performance of the created embeddings, we proposed gating layers that gated only the information that needs to be newly extracted from the embedding vector based on the information extracted from the embeddings so far. And it was proved by experiments that the proposed structure shows high restoration performance in sentence restorations. In addition, it has been shown experimentally that embeddings with minimal information loss show high performance in downstream tasks where information loss is advantageous such as passage retrieval.

However, in this paper, we focused on how to restore sentence-level embeddings to the original text, and we did not study the encoder structure that can create embeddings that contain a lot of information with little loss of information. Therefore, we plan to study the effective encoder structure and objective for this purpose. In this research, information loss was minimized by using an objective that restores the lexical representation, and further research is needed to improve the semantics of embeddings. nally, in order to use the embedding generated in this way in various natural language processing, we plan to study the method of effectively storing information and the structure of referencing and using the stored information.

References

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

Steven Bird and Edward Loper. 2004. NLTK: The natural language toolkit. In *Proceedings of the ACL Interactive Poster and Demonstration Sessions*, pages 214–217, Barcelona, Spain. Association for Computational Linguistics.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Daniel Cer, Yinfei Yang, Sheng-yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St. John, Noah Constant, Mario Guajardo-Cespedes, Steve Yuan, Chris Tar, Brian Strope, and Ray Kurzweil. 2018. Universal sentence encoder for English. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 169–174, Brussels, Belgium. Association for Computational Linguistics.

Alexis Conneau, Douwe Kiela, Holger Schwenk, Loïc Barrault, and Antoine Bordes. 2017. Supervised learning of universal sentence representations from natural language inference data. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 670–680, Copenhagen, Denmark. Association for Computational Linguistics.

Yann N. Dauphin, Angela Fan, Michael Auli, and David Grangier. 2017. Language modeling with gated convolutional networks. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 933–941. PMLR.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

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

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6769–6781, Online. Association for Computational Linguistics.

Ryan Kiros, Yukun Zhu, Russ R Salakhutdinov, Richard Zemel, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Skip-thought vectors. In *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc.

Nikita Kitaev, Lukasz Kaiser, and Anselm Levskaya. 2020. Reformer: The efficient transformer. In *International Conference on Learning Representations*.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. 2019. Natural questions: A benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:452–466.

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.

Yang Liu and Mirella Lapata. 2019. Hierarchical transformers for multi-document summarization. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5070–5081, Florence, Italy. Association for Computational Linguistics.

Joshua Maynez, Shashi Narayan, Bernd Bohnet, and Ryan McDonald. 2020. On faithfulness and factuality in abstractive summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1906–1919, Online. Association for Computational Linguistics.

Deepak Narayanan, Mohammad Shoeybi, Jared Casper, Patrick LeGresley, Mostofa Patwary, Vijay Korthikanti, Dmitri Vainbrand, Prethvi Kashinkunti, Julie Bernauer, Bryan Catanzaro, Amar Phanishayee, and Matei Zaharia. 2021. Efficient large-scale language model training on gpu clusters using megatron-lm. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and*

Analysis, SC '21, New York, NY, USA. Association for Computing Machinery.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

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.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992, Hong Kong, China. Association for Computational Linguistics.

Tobias Rohde, Xiaoxia Wu, and Yinhan Liu. 2021. Hierarchical learning for generation with long source sequences. *CoRR*, abs/2104.07545.

Stephen Roller, Emily Dinan, Naman Goyal, Da Ju, Mary Williamson, Yinhan Liu, Jing Xu, Myle Ott, Kurt Shuster, Eric Michael Smith, Y-Lan Boureau, and Jason Weston. 2020. Recipes for building an open-domain chatbot. *CoRR*, abs/2004.13637.

Aurko Roy, Mohammad Saffar, Ashish Vaswani, and David Grangier. 2021. Efficient Content-Based Sparse Attention with Routing Transformers. *Transactions of the Association for Computational Linguistics*, 9:53–68.

Kurt Shuster, Spencer Poff, Moya Chen, Douwe Kiela, and Jason Weston. 2021. Retrieval augmentation reduces hallucination in conversation.

Yi Tay, Dara Bahri, Liu Yang, Donald Metzler, and Da-Cheng Juan. 2020. Sparse sinkhorn attention. *CoRR*, abs/2002.11296.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Ben Wang and Aran Komatsuzaki. 2021. GPT-J-6B: A 6 Billion Parameter Autoregressive Language Model. `https://github.com/kingoflolz/mesh-transformer-jax`.

Sinong Wang, Belinda Z. Li, Madian Khabsa, Han Fang, and Hao Ma. 2020. Linformer: Self-attention with linear complexity. *CoRR*, abs/2006.04768.

Chuhan Wu, Fangzhao Wu, Tao Qi, and Yongfeng Huang. 2021. Hi-transformer: Hierarchical interactive transformer for efficient and effective long document modeling. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 848–853, Online. Association for Computational Linguistics.

Wenhan Xiong, Xiang Lorraine Li, Srinivasan Iyer, Jingfei Du, Patrick Lewis, William Yang Wang, Yashar Mehdad, Wen-tau Yih, Sebastian Riedel, Douwe Kiela, and Barlas Oğuz. 2021. Answering complex open-domain questions with multi-hop dense retrieval. *International Conference on Learning Representations*.

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

Rowan Zellers, Ari Holtzman, Hannah Rashkin, Yonatan Bisk, Ali Farhadi, Franziska Roesner, and Yejin Choi. 2019. Defending against neural fake news. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Jingqing Zhang, Yao Zhao, Mohammad Saleh, and Peter Liu. 2020. PEGASUS: Pre-training with extracted gap-sentences for abstractive summarization. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 11328–11339. PMLR.

Xingxing Zhang, Furu Wei, and Ming Zhou. 2019. HIBERT: Document level pre-training of hierarchical bidirectional transformers for document summarization. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5059–5069, Florence, Italy. Association for Computational Linguistics.

Yuyu Zhang, Ping Nie, Arun Ramamurthy, and Le Song. 2021. Answering any-hop open-domain questions with iterative document reranking. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '21, page 481–490, New York, NY, USA. Association for Computing Machinery.

A Hyper Parameters Settings for Restoration

Table 6 shows the hyperparameters of the model and optimizer when learning the sentence restoration.

Encoder & Decoder		Optimizer & Generation	
name	value	name	value
d_{model}	512	algorithm	AdamW
number of attention heads	8	learning rate	1e-3
number of attention layers	6	adam epsilon	1e-8
$d_{feedforward}$	2048	weight decay	1e-2
drop out rate	0.1	scheduling	linear
activation for feed forward	relu	warm up	Y
epsilon for layer normalization	1e-6	warm up rate	0.1
max positional embedding size	512	number of beams	4
initialize factor	1.0	early stopping	Y
positional embedding type	relative bucket embeddings	top k	50
positional bucket size	32	top p	50

Table 6: hyper-parameters for training sentence restoration

B Hyper Parameters Settings for Retrieval

Table 7 shows the hyperparameters when learning the passage retrieval.

name	value
batch size	128
epochs	40
optimizer	AdamW
learning rate	1e-3
adam epsilon	1e-8
weight decay	0
scheduling	linear
warm up	Y
warm up rate	0.2
max length for query	70
max length for context	350
number of positive context per sample	1
number of negative context per sample	1

Table 7: hyper-parameters for training passage retrieval

C Full Restoration Performance

Table 8 shows all the restoration performance according to the experimental configuration, the method used to create the embedding vector, and the decoder type.

		classifcation token				mean pooling			
# sents	decoder	PPL	R-1	R-2	R-L	PPL	R-1	R-2	R-L
1	(a) 6 layers from pre-trained model + 0 additional layers								
	input	6.178	9.87	0.79	8.09	1.16	93.37	82.93	89.72
	cross	6.10	7.09	0.19	6.24	1.10	95.14	87.80	92.76
	gating	6.04	11.21	0.55	8.21	1.04	97.76	94.63	96.94
	(b) 6 layers from pre-trained model (freeze) + 0 additional layers								
	input	1.79	13.33	0.75	9.53	2.24	65.99	34.45	50.96
	cross	6.22	12.29	0.78	9.30	2.04	67.97	37.85	54.00
	gating	6.16	11.13	0.29	8.47	1.93	70.54	40.83	56.81
	(c) 6 layers from pre-trained model + 3 additional layers (random initialization)								
	input	6.18	13.32	0.75	9.53	1.15	92.63	83.34	89.63
	cross	6.10	9.95	0.21	8.31	1.12	94.13	86.26	91.62
	gating	6.04	10.81	0.56	8.07	1.03	**98.32**	**96.30**	**97.91**
	(d) 6 layers from pre-trained model (freeze) + 3 additional layers (random initialization)								
	input	6.30	11.86	0.77	8.84	1.34	84.77	69.68	81.12
	cross	6.22	11.21	0.55	8.21	1.29	87.18	73.07	83.79
	gating	6.16	9.88	0.58	7.57	1.09	**95.95**	**91.07**	**95.04**
3	(a) 6 layers from pre-trained model + 0 additional layers								
	input	8.13	13.33	0.48	11.08	2.33	58.98	23.10	40.36
	cross	8.04	13.14	0.26	9.55	1.83	64.86	30.42	47.79
	gating	7.90	18.41	1.14	12.72	1.49	70.79	43.06	58.97
	(b) 6 layers from pre-trained model (freeze) + 0 additional layers								
	input	8.33	12.70	0.07	10.45	4.88	43.60	12.08	24.60
	cross	8.21	14.17	0.34	10.85	4.44	45.37	12.87	25.07
	gating	8.08	14.80	0.79	10.86	4.09	47.52	13.81	25.99
	(c) 6 layers from pre-trained model + 3 additional layers (random initialization)								
	input	8.14	14.32	0.32	11.36	2.31	54.43	21.22	39.01
	cross	8.04	14.48	0.79	10.88	1.89	63.08	29.25	46.87
	gating	7.91	14.67	0.42	11.10	1.37	**72.11**	**50.45**	**64.16**
	(d) 6 layers from pre-trained model (freeze) + 3 additional layers (random initialization)								
	input	8.34	11.20	0.13	9.70	2.96	51.82	18.70	38.18
	cross	8.22	15.07	0.23	11.76	2.48	59.00	24.39	44.09
	gating	8.09	16.81	1.11	11.98	1.75	**67.14**	**39.97**	**58.43**
5	(a) 6 layers from pre-trained model + 0 additional layers								
	input	8.80	11.98	0.24	10.69	3.60	49.63	13.45	28.19
	cross	8.67	15.14	0.87	12.53	2.75	49.63	13.45	28.19
	gating	8.53	11.19	0.21	8.85	2.25	55.36	18.54	35.98
	(b) 6 layers from pre-trained model (freeze) + 0 additional layers								
	input	9.02	13.98	0.09	12.43	6.30	38.24	8.87	20.48
	cross	8.87	13.26	0.21	11.46	5.80	41.25	9.63	21.00
	gating	8.74	11.46	0.12	10.12	5.39	43.66	10.60	21.79
	(c) 6 layers from pre-trained model + 3 additional layers (random initialization)								
	input	8.80	4.71	0.09	4.42	3.36	46.57	12.34	28.54
	cross	8.66	16.96	0.80	12.30	2.80	52.35	15.09	31.28
	gating	8.54	7.42	0.29	6.15	2.08	**52.82**	**18.91**	**36.77**
	(d) 6 layers from pre-trained model (freeze) + 3 additional layers (random initialization)								
	input	9.02	8.02	0.30	7.38	4.19	45.31	11.46	27.65
	cross	8.87	12.02	0.34	10.80	3.50	51.30	14.58	31.00
	gating	8.75	17.16	1.25	11.79	2.76	**52.38**	**17.92**	**36.83**

Table 8: The restoration performance according to the experimental configuration, the method used to create the embedding vector, and the decoder type

D Retrieval Performance of Proposed Model

Table 9 shows the retrieval performance of proposed model according to configurations.

	# sentences	# additional layers	R@1	R@5	R@20	R@100
random initialize		0	14.77	32.68	49.58	67.12
freeze	1	0	21.50	44.11	63.61	78.39
freeze	3	0	21.43	43.96	62.56	77.71
freeze	5	0	21.18	43.61	62.18	77.67
grad	1	0	**24.34**	**47.49**	**64.33**	**78.34**
grad	3	0	22.29	45.05	63.09	78.34
grad	5	0	22.18	45.08	63.09	77.88
random initialize		3	16.88	37.90	55.73	72.37
freeze	1	3	**26.92**	**52.54**	**70.30**	**83.32**
freeze	3	3	24.97	50.02	68.70	82.29
freeze	5	3	25.05	49.56	68.46	82.13
grad	1	3	21.53	45.97	64.07	78.05
grad	3	3	20.97	44.83	63.13	77.82
grad	5	3	22.41	45.13	63.61	78.30

Table 9: Passage retrieval performance in natural questions according to experimental configuration and sentence length

E Performance on Various Sentence level NLP tasks

Table 10 shows the performance of various sentence level downstream tasks when using the sentence embedding of the proposed model.

			GLUE				
			MNLI	QNLI	WNLI	MRPC	QQP
	# sentences	# additional layers	Accuracy	Accuracy	Accuracy	Accuracy	Accuracy
random initialize		0	74.91	80.82	58.33	75.00	**88.81**
freeze	1	0	75.58	81.68	52.78	74.51	88.43
freeze	3	0	75.48	81.66	37.50	**77.21**	88.47
freeze	5	0	**75.58**	**81.92**	55.56	74.26	88.32
grad	1	0	72.38	80.33	56.94	71.81	88.69
grad	3	0	72.34	80.56	**58.33**	74.26	88.69
grad	5	0	72.41	81.28	56.94	73.04	88.50
random initialize		0	74.93	78.53	52.78	74.26	89.89
freeze	1	3	**75.74**	81.97	50.00	71.57	89.96
freeze	3	3	75.73	82.27	55.56	72.79	**90.01**
freeze	5	3	75.69	**82.65**	45.83	73.53	89.96
grad	1	3	72.47	79.83	56.94	72.79	89.04
grad	3	3	72.26	80.38	52.78	**75.25**	89.12
grad	5	3	72.10	80.22	**56.94**	74.26	89.11

			GLUE	SSTDataset	TREC	
			SST2	SSTDataset	Coarse	Fine
	# sentences	# additional layers	Accuracy	Accuracy	Accuracy	Accuracy
random initialize		0	91.28	85.42	**97.02**	**85.91**
freeze	1	0	**91.74**	**86.05**	96.83	85.32
freeze	3	0	91.17	85.96	96.03	85.71
freeze	5	0	91.63	85.96	96.23	83.93
grad	1	0	86.93	77.90	93.85	78.17
grad	3	0	87.84	78.08	94.25	80.16
grad	5	0	87.96	79.17	94.84	81.15
random initialize		0	92.09	85.78	97.02	**92.46**
freeze	1	3	92.55	85.69	96.83	89.48
freeze	3	3	**92.55**	85.33	**97.22**	91.47
freeze	5	3	91.97	**86.50**	96.43	91.67
grad	1	3	87.16	76.54	92.66	83.13
grad	3	3	88.19	77.45	94.84	84.13
grad	5	3	88.76	78.17	94.84	84.72

Table 10: Performance of various sentence level downstream tasks when using the sentence embedding of the proposed model

F Restored Samples

This section shows samples restored by a model trained on sentence restoration (No cherry-picking). In the result of 5 sentences, in the sentence generated by the cross decoder, parts of the sentence such as subject and object were mixed. In sentences generated by the gating decoder, it is rare that parts are mixed. In Table 13, it can be seen that the text generated by the cross encoder is a jumble of information from 5 sentences.

origin	1	Was it a surprise to you that you were given the arts and culture position?
	2	No, there is no surprise when you are a cadre of the ANC because you are deployed anywhere.
	3	You are given a five-year contract to do a portfolio and when you are finished, you wait for another one.
	4	At no stage do you have a say.
	5	What qualities do you bring to the position?
		gating decoder
		1 sentence
restored	1	Was it a surprise to you that you were given the arts and culture position?
		3 sentences
restored	1	Was it a surprise to you that you were given the arts and culture position?
	2	No, there is no surprise when you are a cadre of the ANC because you are deployed overseas.
	3	You are given a five-year contract to do a portfolio and when you (are) finish, you are waiting for another.
		5 sentences
restored	1	Was it a surprise to you that you were given the arts and culture culture?
	2	No, there is no surprise when you are a candidate of the ANC because you are deployed anywhere.
	3	You are given a four-year contract to do a portfolio and when you (are) finish(ed), you are no longer looking for one.
	4	At one stage did you have a capabilities?
	5	What does the message bring to you?
		cross decoder
		1 sentence
restored	1	Was it a surprise to you that you were given the arts and culture position?
		3 sentences
restored	1	Was it a surprise to you when you were given the arts and culture culture?
	2	No, there is no surprise that you are a part of the ANC because you are deployed there.
	3	You are paid a five-year contract when you are ready to do a portfolio and finish another, for five years.
		5 sentences
restored	1	Was it a surprise to you that there was no talent or culture when you were awarded the ANC?
	2	No, you are a part of the arts department.
	3	You are given that you are ready to finish a five-year contract when you are awarded a position and do not finish until a year.
	4	At one stage, do you have another role?
	5	What do you do for the ANC?

Table 11: A sample in which embedding is restored to origin text according to the length of the input text. Blue text means a part different from the original text, and red text means a part omitted from the original text.

origin	1	Occasional diarrhea is a common occurrence.
	2	Most people will experience an episode of diarrhea at least once or twice a year that will disappear in a couple of days.
	3	Luckily, there are many foods to eat that may help a person reduce the symptoms of diarrhea.
	4	There are also some foods to avoid when dealing with a bout of diarrhea, and some additional home care tips to consider.
	5	Anyone who is experiencing persistent diarrhea should see a doctor, as a person may become dehydrated over time.
gating decoder		
1 sentence		
restored	1	Occasional diarrhea is a common occurrence.
3 sentences		
restored	1	Occasional diarrhea is a common occurrence.
	2	Most people will experience an episode of diarrhea at least twice or twice a year that will disappear in a couple of days.
	3	Luckily, there are many foods to eat that may help a person reduce the symptoms of diarrhea.
5 sentences		
restored	1	Occupy diarrhea is a common occurrence.
	2	Most people will experience an episode of diarrhea at least once a month or two that will disappear in a week.
	3	Fortunately, there are plenty of ways to eat a food that may help eliminate the symptoms.
	4	There are also some symptoms of diarrhea to avoid eating with a side dish, and some regular food tips that you should consider.
	5	Anyone experiencing chronic diarrhea will be referred to as a woman, but you have a medical problem before.
cross decoder		
1 sentence		
restored	1	Occasional diarrhea is a common occurrence
3 sentences		
restored	1	Otago occurrences is an uncommon problem.
	2	Most people will experience (an episode of) a diarrhea of at least one day or two during a month that will disappear in less than a month.
	3	Fortunately, there are many ways to eat foods that can help (a person reduce) the symptoms of a person.
5 sentences		
restored	1	Occupied diarrhea is a frequent issue.
	2	Many people will experience a severe diarrhea at least once a week 2014 and that may occur in some cases of diarrhea.
	3	Here are a few things that will stop you to consume more of the food to avoid.
	4	There are also a few cases of diarrhea, while people can experience a side effect to avoid experiencing chronic diarrhea.
	5	If an individual is experiencing chronic diarrhea or diarrhea, some people are able to do a handover after that.

Table 12: A sample in which embedding is restored to origin text according to the length of the input text. Blue text means a part different from the original text, and red text means a part omitted from the original text.

origin	1	Two bedrooms home on a corner lot.
	2	Two car detached garage.
	3	Nice covered front porch.
	4	Seller will not complete any repairs to the subject property, either lender or buyer requested.
	5	The property is sold in AS IS condition.
gating decoder		
1 sentence		
restored	1	Two bedrooms home on a corner lot.
3 sentences		
restored	1	Two bedrooms home on a corner lot.
	2	Two car detached garage.
	3	Nice covered front porch.
5 sentences		
restored	1	Two bedroom home on a corner lot.
	2	Two detached car garage.
	3	Nice covered front porch.
	4	Seller will not complete any repairs to the (subject) property, either insured buyer or seller.
	5	The property is listed in ASOLD condition.
cross decoder		
1 sentence		
restored	1	Two bedrooms home on a corner lot.
3 sentences		
restored	1	Two bedroom homes on a corner lot.
	2	Two car detached garage.
	3	Nice covered front porch.
5 sentences		
restored	1	Two car garage on a corner lot.
	2	Two covered covered porch.
	3	Sony front porch.
	4	Nice covered garage will not return any repairs to the seller, either buyer or seller.
	5	The property is listed in ASOLD condition.

Table 13: A sample in which embedding is restored to origin text according to the length of the input text. Blue text means a part different from the original text, and red text means a part omitted from the original text.