SYNAPSE: Summarization & Answering with Precision via Scalable Encoding

Yajat Rangnekar
2023114008

Manas Mittal
2024121003

George Rahul
2024121013

Abstract

Standard Transformer models are computationally expensive and cannot process long documents like books, legal texts, or in-depth reports in their entirety. This limits their use in comprehensive understanding tasks. We would like to design, build, and evaluate a hierarchical NLP system that can process arbitrarily long documents efficiently. The system will be capable of performing both high-level "gist" tasks (like summarization) and fine-grained "detail" tasks (like question answering) while maintaining factual consistency with the source text. The code for the same can be found here

1 Introduction

The central idea of this project is to develop **SYNAPSE** (Summarization & Answering with Precision via Scalable Encoding), a novel hierarchical framework designed to overcome the computational limits of standard Transformer models when processing long-form documents.

The current state-of-the-art architectures, while powerful, are constrained by the quadratic complexity in their self-attention mechanism (Vaswani et al., 2023), making them impractical for document-scale analysis. SYNAPSE addresses the quadratic bottleneck by adopting a multi-stage, "divide-and-conquer" approach that enables both high-level comprehension and fine-grained factual extraction within a single unified system.

The proposed solution operated via a three-stage pipeline:

1. **Scalable Encoding:** The system first takes a long document as input and parses it into a sequence of smaller, semantically coherent text chunks. It then employs `BGE` to compress the chunk into a meaningful singular vector representation, similar to NextLevelBERT (Czinczoll et al., 2024), which helps in learning a dense and meaningful vector representation (embedding) for each chunk. This initial stage effectively compresses the long document into a computationally tractable sequence of high information.

2. **Task-Adaptive Processing:** The core innovation of SYNAPSE lies in its dual-pathway approach to processing this sequence of embeddings:

   (a) For **Summarization**, which requires a global understanding of the document, the system leverages the entire sequence of chunk embeddings as a high-level abstractive representation.

   (b) For **Answering with Precision**, where factual accuracy is paramount, the system implements an approach similar to Retrieval-Augmented Generation (RAG) (Lewis et al., 2021). A user question is embedded and used to perform a rapid similarity search over the document's chunk embeddings. The system then retrieves the original raw text of the most relevant chunks, providing verifiable source material for the final answer.

3. **Unified Text Generation:** A single generated model trained **end-to-end** (both the encoder and the decoder), built on a unified text-to-text framework like T5 (Raffel et al., 2023) acts as the final output layer. It interprets task-specific prefixes, allowing it fluidly switch between summarization and precise-answering.

By synthesizing these techniques, SYNAPSE aims to create a system that is both scalable for long documents and factually reliable. Its key contribution is a unified framework that provided broad summarization and precise answering, mitigating the risk of hallucinations by grounding responses in the source text.

2 Related Works

2.1 Transformers and Attention

Transformer-based models are highly effective for natural language processing, but they face quadratic complexity challenges when scaling to long sequences (Vaswani et al., 2023). To address representation learning, our scalable encoding strategy draws inspiration from Transformer-based Denoising Autoencoders, which have demonstrated strong performance in producing robust sentence and passage embeddings (Wang et al., 2021). In the answering pathway, we align with the Retrieval-Augmented Generation paradigm, where retrieval improves factual accuracy (Lewis et al., 2021). Finally, our framework takes cues from unified text-to-text models like T5, which show that a single generative system can tackle diverse NLP tasks through task-specific prefixes (Raffel et al., 2023).

2.2 Sentence Embeddings & Semantic Representations

The foundation of SYNAPSE's encoding mechanism relies heavily on advances in sentence embedding techniques. Sentence-BERT (SBERT) revolutionized the field by fine-tuning BERT using siamese networks to produce semantically meaningful sentence embeddings that can be compared using cosine similarity (Reimers and Gurevych, 2019). This breakthrough enabled efficient search and clustering at scale, moving beyond token-level representations to capture sentence-level semantics. Improving upon this foundation by handling diverse granularities, newer models such as BGE (Chen et al., 2024) demonstrated that self-knowledge distilled models could achieve competitive performance while maintaining computational efficiency via efficient batching. The `all-MiniLM` family of models, particularly `all-MiniLM-L6-v2`[1], have also demonstrated that smaller, distilled models could achieve competitive performance while maintaining computational efficiency. These compact models have become widely adopted for document chunking and retrieval tasks due to their favourable balance between speed and semantic understanding.

[1]https://huggingface.co/sentence-transformers/all-MiniLM-L6-v2

2.3 Semantic Chunking & Graph Based Segmentations

Traditional document chunking strategies rely on fixed-size windows or simple heuristics, which often split semantically cohesive units. Workarounds such as overlap between chunks have been used to continue the context of the previous chunks. Recent work has explored most sophisticated approaches that leverage sentence embeddings to identify natural boundaries in documents. Graph-based methods represent documents as networks where nodes correspond to sentences and edges encode semantic similarity (Mihalcea and Tarau, 2004). Community detection algorithms applied to these graphs can identify clusters of semantically related sentences, enabling more intelligent chunking that respects topic boundaries and discourse structure. This approach aligns with SYNAPSE's goal of creating semantically coherent chunks that preserve contextual information while reducing sequence length.

2.4 Long Document Summarization Models

The summarization component of SYNAPSE draws from recent advances in abstractive summarization. BART (Bidirectional and Auto-Regressive Transformers) combines the strengths of BERT's bidirectional encoder with GPT's autoregressive decoder, demonstrating exceptional performance on summarization benchmarks through its denoising pretraining objective (Lewis et al., 2019). T5 takes this further with its unified text-to-text framework, treating every NLP task as a seq2seq problem and showing that a single model can excel across diverse tasks including summarization (Raffel et al., 2023). For extremely long documents that exceed typical context windows, Longformer-Encoder-Decoder (LED) introduced sparse attention patterns that scale linearly with sequence length, enabling direct processing of documents with thousands of tokens (Beltagy et al., 2020). These models demonstrate various strategies for handling long-range dependencies in summarization tasks.

2.5 Next Level BERT

Recent advances in language modeling have significantly improved representation learning, but processing long documents remains challenging due to the quadratic complexity of self-attention in Transformer-based models. To address this, several works have explored operating on higher-level representations rather than individual tokens. For instance, NextLevelBERT is a masked language model trained to predict the embeddings of masked text chunks instead of tokens, enabling it to capture both local and global context efficiently (Czinczoll et al., 2024). This approach has been shown to perform well on tasks such as semantic textual similarity, long-document classification and multiple-choice question-answering, often outperforming much larger embedding models when fine-grained token-level detail is not required. Such hierarchical or chunk-based modeling strategies are increasingly adopted in modern NLP systems to better handle longer sequences while maintaining computational efficiency.

2.6 Retrieval-Augmented Generation (RAGs)

The RAG paradigm addresses a critical weakness of pure generative models, the tendency to hallucinate or produce unsupported claims. By retrieving relevant passages before generation, RAG systems ground their outputs in verifiable source material (Lewis et al., 2021). Recent extensions have explored dense passage retrieval, where both queries and documents are embedded in a shared semantic space, enabling efficient similarity-based retrieval (Karpukhin et al., 2020). SYNAPSE adopts this paradigm but integrates it more tightly into the architecture, using the same embeddings for both summarization context and precision retrieval, creating a unified representation that serves dual purposes.

3 Methodology

3.1 Semantic Chunking

We implemented a graph-based semantic chunking approach as showing in Algorithm 1. In this method, each sentence in the document is represented as a node, and edges are added between nodes if their cosine similarity exceeds a predefined threshold τ. The resulting similarity matrix forms an undirected graph. We then identify connected components in the graph, with each connected component corresponding to a single chunk. All the experiments conducted below are done on a subsample of the dataset and not on the whole dataset.

3.1.1 MiniLM

We initially employed the `MiniLM-L6-v2` model to generate sentence embeddings, which were subsequently used to compute similarity scores between

Algorithm 1 Graph-Based Semantic Chunking

```
Require: Document $D = \{s_1, \ldots, s_n\}$, similarity threshold $\tau$
Compute cosine similarity for all sentence pairs to form a similarity matrix
Construct an undirected graph $G$ where each sentence is a node
Add an edge between nodes $s_i$ and $s_j$ if $\text{CosSim}(s_i, s_j) \geq \tau$
Identify all connected components in $G$
Each connected component forms a chunk; add it to $\mathcal{C}$
return $\mathcal{C}$
```

textual units. From the results presented in Table 1, we determined that the optimal similarity threshold is $\tau = 0.5$.

When $\tau < 0.5$, the entire document is treated as a single chunk, effectively negating the purpose of semantic segmentation. Conversely, when $\tau > 0.5$, the dataset fragments into approximately 3,000 chunks, with each chunk corresponding roughly to a single sentence. This results in excessive granularity, where each chunk represents an individual sentence rather than a coherent semantic group.

However, this approach exhibits a critical limitation that is discussed in Section 3.2 and 5.1

τ	Chunks	Avg	Med	Max
0.1	1	3847.000	3847	3847
0.2	1	3847.000	3847	3847
0.3	1	3847.000	3847	3847
0.4	126	30.532	1	3715
0.5	1447	2.659	1	2070
0.6	3012	1.277	1	200
0.7	3609	1.066	1	45
0.8	3775	1.019	1	11
0.9	3818	1.008	1	11

Table 1: Graph-Based Chunking: chunk statistics by threshold τ.

3.2 BGE

To address the limitations of MiniLM, we adopted the `BGE-large-en-v1.5` model (Chen et al., 2024). This model overcomes a fundamental issue overlooked in the previous approach. MiniLM-L6-v2 is trained with a maximum input length of only 128 tokens, which constrains its ability to encode longer, semantically rich chunks. Additionally, its embedding dimensionality is limited to 384.

In contrast, BGE-large-en-v1.5 supports sequences up to 8,000 tokens and produces embeddings of 1,024 dimensions. As a result, it can encode lengthy chunks more effectively, capturing richer contextual relationships and offering significantly more expressive representations of the textual content.

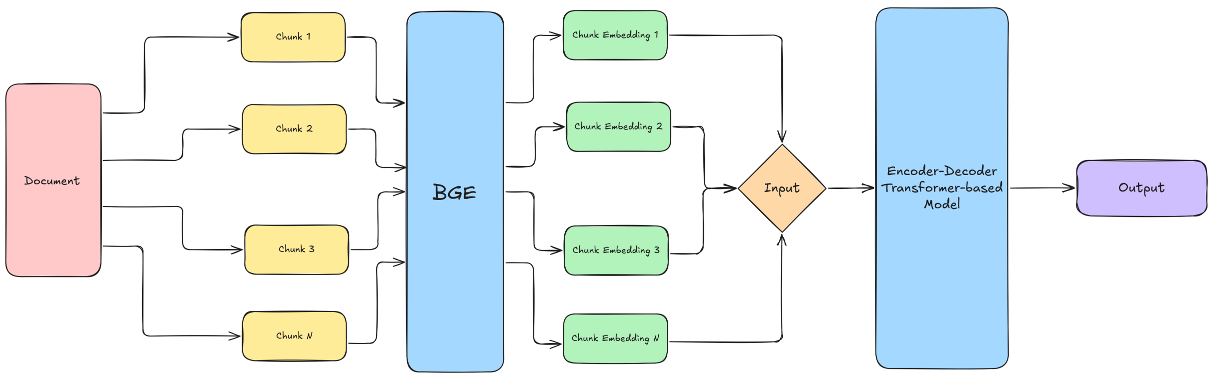

Figure 1: Overview of our proposed pipeline. The system converts the document into semantic chunks which are compressed into chunk embeddings before passing them to the transformer for the specified task.

From the results presented in Table 2, we observe that the optimal threshold is $\tau = 0.7$. This value yields approximately 2.9K chunks about half the number obtained at $\tau = 0.9$, which produces overly fine-grained segmentation. Thresholds greater than 0.7 result in a rapid increase in the number of chunks, leading to excessive fragmentation, while thresholds below 0.7 produce too few chunks, causing semantically distinct sections to merge.

To further refine chunk selection, we incorporated an outlier detection mechanism to identify and exclude extreme chunk sizes. Specifically, we removed chunks falling below the 2nd percentile and above the 98th percentile of the size distribution. For $\tau = 0.7$, the proportion of outliers was minimal, reinforcing our conclusion that this threshold provides the most balanced segmentation.

Moreover, the embeddings generated by the BGE model exhibit noticeably higher quality. The expanded input length allows it to capture a broader semantic context, effectively grouping sentences that are thematically related, which was not possible with the MiniLM model.

3.3 Chunk Embeddings

As discussed in the previous sections, our approach begins by dividing each document into semantically coherent chunks (see Section 3.1). Once the text is segmented, we obtain embeddings for each chunk.

τ	**Chunks**	**Avg**	**Med**	**Max**	**Outlier%**
0.1	1	4647.000	4647	4647	0.0
0.2	1	4647.000	4647	4647	0.0
0.3	1	4647.000	4647	4647	0.0
0.4	1	4647.000	4647	4647	0.0
0.5	1	4647.000	4647	4647	0.0
0.52	1	4647.000	4647	4647	0.0
0.54	4	1161.750	1	4644	0.0
0.56	26	178.731	1	4622	0.0
0.566	38	122.289	1	4609	2.6
0.57	45	103.267	1	4602	2.2
0.573	52	89.365	1	4595	1.9
0.576	61	76.180	1	4586	1.6
0.58	75	61.960	1	4570	1.3
0.6	214	21.715	1	4427	0.5
0.63	651	7.138	1	3968	0.3
0.67	1836	2.531	1	2709	0.2
0.7	2948	1.576	1	1354	0.6
0.73	3778	1.230	1	372	0.8
0.76	4253	1.093	1	87	0.4
0.8	4496	1.034	1	34	1.5
0.9	4612	1.008	1	10	0.3

Table 2: Threshold analysis results showing chunk statistics by τ.

Employing a transformer-based sequence-to-sequence architecture for this task was deemed infeasible due to both computational limitations and the scarcity of training data. Consequently, we adopted a more efficient and scalable approach: each chunk was first embedded individually, and these embeddings were then encoded and transformed using autoencoders and projection matrices before being fed into the model. Similar to the earlier sections, MiniLM and BGE models (`BGE-large-en-v1.5` and `MiniLM-L6-v2`) were utilized for generating chunk embeddings. The limitations associated with MiniLM, as discussed previously, persist in this context as well.

3.4 Adapter Layers & Full Fine Tuning

We explored two distinct strategies to bridge the embedding space and the model's internal representation: Adapter Layers and Full Finetuning. In our approach, each chunk embedding was treated as a token level input to the model. In other words, instead of individual words serving as tokens, we considered each chunk typically a few sentences as a single token representation.

The Adapter based experiments aimed to map these chunk embeddings to the model's existing token representations without altering the model's core parameters. This allowed us to investigate whether the model could effectively align the chunk level embedding space with its native token level representation through lightweight adaptation.

In contrast, the Full Fine Tuning approach updated all model parameters, effectively compelling the model to internalize and represent this new form of chunk based input directly, rather than merely aligning preexisting spaces. We conducted fine tuning experiments on both pretrained and untrained variants of the model to examine whether prior knowledge played a significant role.

4 Experiments

Most of the experiments have been performed for the task of summarization since embedding-based retrieval mechanisms are quite studied upon (Lewis et al., 2021). Experiments were run for the task of question answering as well, but they lead to not-so satisfactory results, the reasons for the same shall be covered in Section 5.

4.1 Training Setup

We used the Ada nodes from IIIT Hyderabad[2]. The training setup utilized four NVIDIA GTX 1080 Ti GPUs and forty CPU cores.

4.2 Data Setup

NarrativeQA (Kočiský et al., 2017) is a reading comprehension dataset built from books and movie scripts (around 1,567 stories), paired with human-written summaries and question-answer pairs (46,765 pairs). The dataset provides two main evaluation settings: (i) *Summaries Only*, where questions must be answered based on a Wikipedia-style summary of the narrative, and (ii) *Stories Only*, which requires models to operate over the full long-form text, often tens of thousands of words in length. Because of its scale and narrative complexity, NarrativeQA forces models to perform chunk-based retrieval and deep comprehension across lengthy contexts, aligning directly with SYNAPSE's scalable encoding and hierarchical summarization design.

[2]Experiments were conducted on the Ada GPU cluster at IIIT Hyderabad.

4.3 Baselines

We used the following models as baselines to evaluate against our proposed approach: (i) BART, (ii) T5, and (iii) LLaMA. For these baselines, we performed standard inference using the pretrained checkpoints without any fine-tuning. This decision was motivated by several factors. Firstly, the documents in our dataset are considerably long, exceeding the context window of the models. As a result, truncation during preprocessing leads to substantial information loss, making full finetuning ineffective and computationally inefficient. Secondly, due to hardware constraints, the feasibility of large-scale finetuning was reduced. Finally, the documents themselves are highly unstructured and contain a significant amount of irrelevant or noisy content. While such characteristics tend to degrade the performance of standard models, our proposed method is designed to handle unprocessed data more effectively, as its compression mechanism filters out much of the syntactic noise that does not contribute meaningfully to the document semantics.

Metric	BART Base	T5 Base
R-1	0.092	0.021
R-2	0.012	0.002
R-L	0.061	0.018
BS-F1	0.767	0.762

Table 3: Baseline performance of pretrained BART and T5 models. R: ROUGE, BS: BERTScore.

4.4 Mini LM V6 Experiments

Initially, we attempted to chunk the document into semantically coherent groups using `all-MiniLM-L6-v2` to embed the sentences. The output embeddings of the model were of the dimensions 384. An autoencoder was used to compress the given embeddings further to reduce the dimension, thereby creating semantically rich embeddings that can be passed to the model as context. During model training, the low-dimensional latent embeddings were up-projected to the hidden dimension of the target model's token dimensions.

We tested two methods to train the model with the given embeddings: adapter layers, where the embeddings would be transformed into a format the target model can understand and process and full-finetuning, where the embeddings are provided

	BART (Full)			T5 (Full)			LLAMA (Adapter)			T5 (Adapter)		
Metric	**FT**	**Base**	**Diff**	**FT**	**Base**	**Diff**	**Adapt**	**Base**	**Diff**	**Adapt**	**Base**	**Diff**
R-1	0.202	0.073	+176%	0.101	0.055	+85%	0.003	0.172	-99%	0.099	0.054	+84%
R-2	0.005	0.010	-51%	0.004	0.008	-51%	0.000	0.026	-99%	0.002	0.008	-69%
R-L	0.085	0.051	+68%	0.057	0.039	+49%	0.002	0.078	-97%	0.054	0.039	+40%
BS-P	0.738	0.746	-1.1%	0.744	0.766	-2.8%	0.687	0.734	-6.4%	0.737	0.784	-5.9%
BS-R	0.759	0.770	-1.5%	0.758	0.769	-1.4%	0.760	0.781	-2.6%	0.757	0.764	-0.9%
BS-F1	0.748	0.758	-1.3%	0.751	0.767	-2.1%	0.722	0.757	-4.6%	0.747	0.774	-3.5%

Table 4: Comprehensive comparison of fine-tuned and adapter models vs. base models for BART, T5, and LLAMA. FT: Fine-tuned, Adapt: Adapter, Diff: Percentage difference from base model, R: ROUGE, BS: BERTScore, P: Precision, R: Recall. All differences are calculated as ((Model - Base) / Base) × 100%.

to the target model as is and the model learns the new semantically rich representations and how to interpret them to generate the required output.

4.5 BGE

Upon further analysis and experimentation, we deduced that the autoencoder-based architecture a few critical flaw: due to lack of a large enough dataset, the autoencoders trained could not generalise well and create perfectly semantically rich and coherent latent embeddings, the embeddings created were of lower quality and noisy leading to sub-standard generation and outputs. Extensive literature review revealed newer sentence embedding models that were more suited for the given task, such as BGE (Chen et al., 2024). Unlike conventional autoencoders that rely solely on reconstruction loss, BGE leverages large-scale pretraining on both retrieval and generation tasks to produce semantically rich latent embeddings for the input chunks. By replacing the original pipeline with the BGE embedding model, we achieved significantly better performance.

5 Results, Analysis & Limitations

5.1 Mini LM V6 Experiments

Adapter layers were flawed for our specific use case, the results upon using the adapter layers were subpar. One of the possible reasons for the same is that it tries to forcefully map the compressed semantically rich latent representation to a sparse embedding representation; it tries to map the chunk embedding to a word embedding, which leads to a lot of information loss and noise. The model fails to effectively understand and utilize these embeddings.

Finetuning performed better, since the model had to learn to utilize the new semantically rich embeddings to generate the output. However, the core idea of using MiniLM had a few fundamental limitations. Firstly, the embeddings generated by `all-MiniLM-L6-v2` have a dimension restriction of 384, which needs to be up-projected to match the dimensions of the target model being used. Due to lack of substantial training data, the up-projection step is flawed and leads to noisy embeddings being passed to the model, thus hampering the generative capabilities of the model.

5.2 BGE Experiments

In order to overcome the drawbacks of the first approach, we decided to use BGE `BGE-large-en-v1.5`. Unlike `all-MiniLM-L6-v2`, BGE compresses the chunk to embeddings of dimensions 1024. This is a huge improvement in the generation of the chunks and the embeddings, since due to the elimination of the up-projection step, the embeddings have lesser noise and can thus capture more relevant information.

We tested these chunk embeddings on two encoder-decoder models: T5 and BART.

5.3 Improvement Analysis

The table 4 shows huge performance improvements over the vanilla models. Sometimes we do see a gain of upto 1000% in Table 5. This can be attributed because of how poor the baselines are. Because of the limited input length that these vanilla architectures can handle, the models often receive only the boilerplate portion of the scripts as input. Although we attempted to clean the dataset using regular expressions, the inherently unstructured nature of the data (HTML, text,etc.). We did try cleaning up the dataset with regex patterns but, still due to the unstructured nature of the dataset (it contains HTML Pages, text,etc.) still led to the inclusion of unwanted content. Even under ideal conditions, we believe this limitation would persist, since the baselines can only process a small

Training Strategy	**Metric**	**T5-Base**			**BART-Large**		
		Trained	**Base**	**Improv.**	**Trained**	**Base**	**Improv.**
Fully Finetuned (Enc+Dec)	R-1	0.190	0.022	+762.6%	0.213	0.093	+130.4%
	R-2	0.034	0.002	+1435.2%	0.041	0.012	+249.7%
	R-L	0.105	0.018	+497.8%	0.126	0.061	+106.8%
	BS-F1	0.813	0.760	+6.9%	0.810	0.767	+5.6%
Decoder Frozen (Enc only)	R-1	0.100	0.022	+355.2%	0.208	0.093	+125.3%
	R-2	0.016	0.002	+633.7%	0.031	0.012	+164.7%
	R-L	0.064	0.018	+267.0%	0.119	0.061	+95.5%
	BS-F1	0.776	0.760	+2.1%	0.803	0.767	+4.7%
Untrained (No pretrained)	R-1	0.097	0.021	+367.2%	0.239	0.092	+158.7%
	R-2	0.022	0.002	+1004.4%	0.052	0.012	+336.4%
	R-L	0.066	0.018	+271.2%	0.137	0.061	+126.5%
	BS-F1	0.797	0.762	+4.5%	0.785	0.767	+2.4%
Cleaned Dataset (Preprocessed)	R-1	0.199	0.021	+858%	0.177	0.039	+350%
	R-2	0.037	0.001	+2608%	0.031	0.003	+1096%
	R-L	0.111	0.017	+542%	0.109	0.023	+374%
	BS-F1	0.813	0.737	+10.3%	0.809	0.742	+9.0%

Table 5: Comprehensive comparison of T5-Base and BART-Large models across different training strategies. All models trained on chunks. R: ROUGE, BS: BERTScore, Improv.: Percentage improvement over base pretrained model.

fraction of a full movie script making it practically impossible for them to capture the context required for summarization. Thus, any improvement will show a huge gain the metrics and we thus went with human evaluation of a subset of the samples to see if the model is actually performing well.

5.4 Sample Analysis

We did a human evaluation pass over 25 random samples for each of the configurations mentioned in the former sections. It was evident that the model has over fitted since most of the generated outputs have nearly identical opening sentences. However the last few phrases or sentences does look promising. Take Sample A 3 for example. The model does get the grim tone correct. The reference summary talks about how four orphans adapt from an aristocratic lifestyle to that of simple foresters under the character Armitage. The generated summary talks about how a young man who was wealthy turned poor and how he was raised by his aunt. In sample A 9 we can see that the model does identify that the story takes place in Paris while in sample A 12 identifies the story of the movie Thor and relates it to aliens coming to a planet. Though it does goes off track after that.
One thing we did notice is that BART performs worse than T5. This we believe is because T5 is a more modern and a flexible architecture thus, helping the model to adapt better to the new schema of learning. The examples show above should be taken with a grain of salt because there are a lot of samples in which it does goes completely off track or just repeats irrelevant phrases/sentences. We just want to highlight the potential of our solution.

5.5 Dataset as a bottleneck

We do believe that one of the main reasons why we our results are not as good as expected is because of the lack of the data. Next Level BERT (Czinczoll et al., 2024), trained on the “Pile of Books" Dataset. This is contains over 200,000 books in it. On the other hand, we had only 1,000 summaries to train our model with the additional burden of integrating the decoder. Thus, when using a pretrained model, the lack of data prevents the model from fully adapting to the new knowledge mapping. Conversely, in the case of an untrained model, the dataset is insufficient to effectively train the decoder. Therefore, we identify the dataset size and diversity as one of the most critical limitations in our current implementation.

5.6 Adapter vs Full Finetuning

Though we did not try adapters in the BGE embeddings part of the architecture, we fundamentally believe that Adapters are not the way to go in the architecture. In table 4 we can see that the full finetuning has a more important significant impact. This is because we treat entire sentences as single tokens rather than say individual words. Adapters merely attempt to map these representations without the model inherently expecting such input structures, whereas full finetuning allows the model to adapt its internal behavior accordingly.

6 Conclusion and Future Work

We presented SYNAPSE, a hierarchical framework for long-document summarization and question answering. Using graph-based semantic chunking with BGE embeddings, SYNAPSE effectively compresses lengthy texts into semantically rich representations, achieving notable gains over baseline models.

While results are promising, challenges such as limited training data, overfitting, and the unstructured nature of the NarrativeQA dataset remain.

We plan to extend this framework by (1) training on larger and cleaner datasets, (2) training autoencoders on better and larger corpus to facilitate better and more diverse compression strategies. (3) refining adapter layers for chunk-level embeddings, (4) enhancing the Q&A pathway with improved retrieval and reasoning, and (5) exploring adaptive chunking and human evaluation methods to strengthen real-world applicability.

References

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

Jianlv Chen, Shitao Xiao, Peitian Zhang, Kun Luo, Defu Lian, and Zheng Liu. 2024. Bge m3-embedding: Multi-lingual, multi-functionality, multi-granularity text embeddings through self-knowledge distillation. *Preprint*, arXiv:2402.03216.

Tamara Czinczoll, Christoph Hönes, Maximilian Schall, and Gerard de Melo. 2024. Nextlevelbert: Masked language modeling with higher-level representations for long documents. *Preprint*, arXiv:2402.17682.

Vladimir Karpukhin, Barlas Oğuz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen tau Yih. 2020. Dense passage retrieval for open-domain question answering. *Preprint*, arXiv:2004.04906.

Tomáš Kočiský, Jonathan Schwarz, Phil Blunsom, Chris Dyer, Karl Moritz Hermann, Gábor Melis, and Edward Grefenstette. 2017. The narrativeqa reading comprehension challenge. *Preprint*, arXiv:1712.07040.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *Preprint*, arXiv:1910.13461.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2021. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Preprint*, arXiv:2005.11401.

Rada Mihalcea and Paul Tarau. 2004. TextRank: Bringing order into text. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, pages 404–411, Barcelona, Spain. Association for Computational Linguistics.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2023. Exploring the limits of transfer learning with a unified text-to-text transformer. *Preprint*, arXiv:1910.10683.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. *Preprint*, arXiv:1908.10084.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2023. Attention is all you need. *Preprint*, arXiv:1706.03762.

Kexin Wang, Nils Reimers, and Iryna Gurevych. 2021. Tsdae: Using transformer-based sequential denoising auto-encoder for unsupervised sentence embedding learning. *Preprint*, arXiv:2104.06979.

A Reference Samples

Sample 3 (T5 full finetune)

Reference: ... Under Armitage's guidance, the children adapt from an aristocratic lifestyle to that of simple foresters. After Armitage's death, Edward takes charge and the children develop and expand the farmstead, aided by the entrepreneurial spirit of the younger brother Humphrey. They are assisted by a gypsy boy, Pablo, who they rescue ...
Generated: The story is set in the early years of the 19th century and begins with a brief history of the family's life. It tells the story of a young man named John, who was born into slavery at the age of five to a wealthy widower-turned-poverty. He has been raised by his aunt, Mrs. Pearce, for many years as a child, but has never had the opportunity to marry him. At the beginning of the 20th century, he met his future wife, Mary, when she was only seven years old when she became pregnant. When John's father died, John returned to England to pursue a career in her youth. After a

Sample 9 (T5 full finetune)

Reference: ... During his stay in Paris, Yorick is informed that the police inquired for his passport at his hotel. Without a passport at a time when England is at war with France (Sterne traveled to Paris in January 1762, before the Seven Years' War ended), he risks imprisonment in the Bastille. Yorick decides to travel to Versailles where he visits the Count de B**** to acquire a passport. ... Yorick returns to Paris, and continues his voyage to Italy after staying in Paris for a few more days. Along the way he decides to visit ...
Generated: The novel is set in the early years of the 19th century. It tells the story of a young man, who has just returned from a trip to Paris with his wife and two children. The story begins when he discovers that he has been mistreated by his father for not having enough money to pay off his debts. He decides to take a job as a waitress at a small-town hotel. After a few days, however, he finds himself in a situation where he can't afford to pay his rents due to a low-paid position at the hotel. When he comes back, it becomes clear to him that if he

Sample 12 (T5 full finetune)

Reference: In 965 AD, Odin, king of Asgard, wages war against the Frost Giants of Jotunheim and their leader Laufey, to prevent them from conquering the nine realms, starting with Earth. The Asgardian warriors defeat the Frost Giants and seize the source of their power, the Casket of Ancient Winters. ...
Generated: In the early 20th century, a group of aliens from the distant past have been dispatched to destroy a planet. The aliens are being hunted and killed by a mysterious force that has destroyed the planet's solar system. As a result, the Earth's population has fallen to a low point, and the spacecraft is about to be dismantled. They are joined by an army of scientists, led by Captain James T. Kirk, who is in charge of the project. Kirk is assigned the task of destroying the planet on his way out of the galaxy, as well as completing a mission to find a meteor strike on the planet