Hierarchical Architecture for Long Document Processing

Project Area: Document Understanding

Yajat Rangnekar	Manas Mittal	George Rahul
yajat.rangnekar@research.iiit.ac.in	manas.mittal@research.iiit.ac.in	george.rahul@research.iiit.ac.in
2023114008	2024121003	2024121013

***Team Name**: LE and Yajat*

Abstract

Current Transformer-based models face computational limitations when processing long documents due to their quadratic complexity, making them impractical for document-scale tasks like book summarization or comprehensive legal document analysis. We propose a novel hierarchical architecture that addresses this scalability challenge while maintaining both global context and fine-grained detail accessibility.

Our approach implements a three-stage pipeline: (1) semantic chunking using sentence embeddings to create coherent document segments, (2) compression of each chunk into latent representations via autoencoders, and (3) a main model trained on these compressed representations with a dynamic gating mechanism. The key innovation lies in our adaptive inference strategy, where a gating mechanism determines whether to operate on compressed representations (for global understanding tasks like summarization) or decompress chunks back to token-level detail (for precise tasks like question-answering). This design enables processing of arbitrarily long documents while maintaining computational efficiency and factual consistency.

Methodology

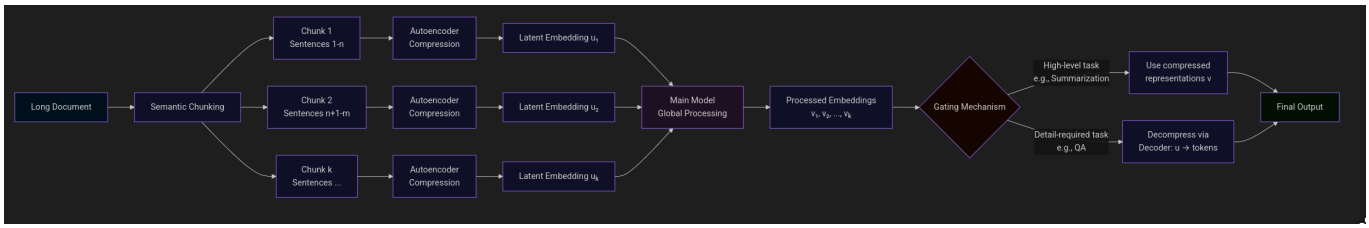

Figure 1: Pipeline of the Proposed Architecture

Chunking

This step is to divide the document into multiple chunks. The sentences in the chunk should be related to each other semantically. To ensure this, we can use SBERT to generate the CLS token of each sentence, and then use cosine similarity to check if the sentences are related or not.

Fine-grained embeddings

For each chunk, we generate embeddings for the tokens in it. These embeddings are then compressed via an autoencoder or its variant to get k latent embeddings. These embeddings represent the ideas in the chunk.

Main Model

The main model is trained on these latent embeddings, so it is trained on the ideas represented by the chunks, and do not need to focus on the fine-grained details of it. Let original latent embedding be u. After training, it will become v. For a particular test of a given task, some chunk embeddings will be more relevant to the output than others.

Inference

We implement a gating mechanism between u and v to determine what should be focused on more — the details of the chunks (ex: QA) vs the idea of the chunk (ex: summarization). Accordingly, the embedding w is passed through the decoder block of the autoencoder to re-generate the embeddings of the tokens of the chunk as needed. These embeddings are then used to generate the final answer.