Context-Aware Extractive Summarization Using Pretrained Models for Long Documents

Sammer Sami Abdulkareem
Mohammad-Reza Feizi-Derakhshi
Computerized Intelligence Systems Laboratory, Department of Computer Engineering, University of Tabriz, Tabriz, 51666-16471, Iran
E-mail: sammer.sami.abdulkareem@gmail.com, mfeizi@tabrizu.ac.ir

Keywords: Text summarization, extractive summarization, long document, pretrained models

Received: January 3, 2026

Recently, significant advancements have been made in models for automatic text summarization, primarily because of the emergence of modern technologies such as transformers and pre-training. However, handling long documents remains a major challenge for most models. Current research addresses this challenge either by designing recurring chained models or by truncating the text to fit the input size. Insufficient attention has been given to preserving the context of the original text and its role in summary generation. We propose a model that captures consecutive sentence contexts and processes them as coherent units. Using an unsupervised summarization pipeline that requires no task-specific fine-tuning, the model produces extractive summaries composed of the most informative sentences from the source document. We evaluate the proposed pipeline on the PubMed and ArXiv long-document benchmarks using ROUGE, achieving ROUGE-1/ROUGE-2/ROUGE-L of 53.0/24.32/47.7 on PubMed and 53.7/20.40/48.4 on ArXiv.

Povzetek: Študija predlaga nenadzorovan ekstraktivni povzemalnik dolgih dokumentov, ki namesto rezanja ali verižnega procesiranja združuje zaporedne stavčne kontekste v koherentne enote in izbere najbolj informativne stavke ob boljšem ohranjanju izvornega konteksta.

1 Introduction

The Internet inundates our days with vast amounts of text content, including social media posts and scientific papers. This growth provides useful information, but it also makes processing and understanding that information harder for people and businesses. Automatic text summarization has become an important tool for managing such data [1]. The goal of this summarization process is to turn long texts into short summaries that retain the most important information [2]. This process remains challenging for long documents such as research papers, technical reports, and scholarly articles because they are so lengthy and complex. Therefore, text summarization remains an active area of study in natural language processing [3].

Automatic text summarization can be divided into two broad categories: extractive and abstractive methods [4]. Abstractive summarization takes a text and produces a shorter version that states the same ideas with different words [5]. Extractive summarization, on the other hand, selects the most crucial sentences in the original text and compiles them to form a brief summary. The abstractive methods tend to generate better summaries and are more likely to be fluent, although they may introduce errors and employ alternative words that reduce precision [6, 7]. According to research, extractive methods are often preferred in human tests for factual preservation, though this advantage varies across domains and evaluation criteria [8].

Traditional summarization approaches include statistical and graph-based methods [9, 10, 11]. The introduction of pretrained Transformer encoders such as BERT [12] substantially improved summarization, but long inputs remain challenging because self-attention is constrained by token budgets and scales poorly with input length [13, 14]. Prior work addresses long documents using truncation [15, 16, 17, 18, 19, 20], sliding windows [21], capacity-aware modeling [13], and hybrid mechanisms [22], while hierarchical formulations provide an alternative for capturing multi-scale dependencies [23].

While the framework leverages established pretrained components, the contribution lies in an unsupervised extractive pipeline for long scientific documents that (i) performs budget-constrained context grouping using semantic continuity and discourse cues, (ii) uses generated candidates as global salience proxies to guide sentence ranking without fine-tuning on PubMed/ArXiv, and (iii) supports key design choices with targeted ablation and sensitivity analyses. In contrast to fixed-window segmentation or whole-document scoring, candidate generation and ranking are performed within context-consistent, token-bounded context groups to better reflect long-document structure.

The aforementioned studies have addressed the limitations of BERT, but they share the common limitation of

dividing input documents into fixed-size sentence chunks. The importance of continuity between successive sentences is ignored, which is crucial for preserving content and maintaining semantic coherence in the final summary. This research introduces a model that considers the content of consecutive sentences in a document to guarantee the comprehensiveness of the context provided in the summary. The enhanced outcomes of our approach emphasize the importance of considering the context within the succession of sentences. Long-document models perform poorly when the context is ignored or truncated.

The goal of this work is to produce faithful extractive summaries of long scientific documents under practical token-budget constraints, without fine-tuning on the target datasets (PubMed/ArXiv). To preserve continuity in long inputs, we first group consecutive sentences into token-bounded, context-consistent units using semantic similarity and discourse markers; to obtain a document-level salience signal without supervision, we generate candidate texts within each group and pool them across the document; we then rank original sentences by semantic similarity to these pooled candidates and apply n-gram blocking to control redundancy before selecting the final sentences in their original order. This design ensures each step serves a specific purpose: context preservation, salience estimation, and redundancy-controlled sentence selection.

2 Related works

As requirements increase, researchers are increasingly interested in automated summarization, a branch of natural language processing (NLP). Researchers have proposed several approaches for both abstractive and extractive summarization over the years. Much prior work has focused on methods for summarizing long documents.

Many studies have attempted to treat the issue of long documents by using some methods like truncating text to keep it in a 512-token requirement [15, 16, 17]. Unsupervised approaches have been suggested and supported by several studies [24, 25, 26, 20]. One example is graph-based models. Several proposed methods [27, 28] rely on classifying sentences as either included in the final summary or excluded. A GRU-RNN encoder has been adopted for neural extraction-based document summarization, as reported in [29]. Redundancy in the summary is reduced using a phrase extractor based on Recurrent Neural Networks (RNNs). Furthermore, the initially identified segments are examined in detail. A study [30] proposes an approach, Attention with CNN–BiGRU, to detect sentence characteristics. The CNN employs intricate techniques that reveal the concealed features of words. Thereafter, a BiGRU model is employed to conduct the summary. The BiGRU is used to identify appropriate relationships within long text sequences [31]. This method enables researchers to identify depression-related signs in tweets using evaluation language.

The extractive summarization method employs long-short term memory (LSTM) models in several studies. To determine which texts should be included or excluded from the final summary, the study [32] uses a CNN as an encoder and an LSTM as a decoder. Another model utilizes a BiLSTM encoder–decoder architecture to identify connections between important phrases and keywords. The SWAP–NET model is also proposed by [33] as a new summarization method. This model ranks sentences using hierarchical transformers and phrase-level attention. Studies have also shown that this method produces highly accurate summaries [17].

The BERT model is considered the foundation for many extractive summarization algorithms. The work by [19] examines an extractive summarization approach, a semantic text technique based on BERT, to identify semantically plausible summaries. In the study of [34], two models are used to evaluate the range and salience of texts. Their method, which depends on an extensive redundancy-based framework, is good at scoring and choosing text portions and maintaining a balance to generate the final summary. In the study of [23], an approach that integrates data from sentence embeddings and document representations is proposed. This approach combines top-down and bottom-up mechanisms to maintain representations up to date and manipulates token attention across multiple levels, namely, random, local, and global.

More recently, several studies have revisited long-document summarization using stronger pre-trained encoders and, in some cases, large language models. Wang et al. [35] propose an extractive model for long news and scientific articles that injects local topic information and document-level hierarchical structure into a Longformer-based encoder. Han et al. [36] design a topic-model-based extractor with sentence-level features and a dynamic memory unit to better capture global themes in long texts, while Onan and Alhumyani [37] combine fuzzy topic modeling with BERT in the FuzzyTP-BERT framework for extractive summarization. In the biomedical domain, Xie et al. [38] incorporate domain knowledge into pre-trained language models to improve extractive summarization of scientific articles, and Kirmani et al. [39] introduce a semantic summarizer tailored to biomedical literature. Complementary to these encoder-based models, Chen et al. [40] demonstrate that large language models equipped with hierarchical input segmentation and chain-of-thought prompting can improve long-document summarization across scientific and biomedical benchmarks.

Table 1 summarizes representative long-document summarization designs. Many strong long-document systems rely on target-dataset fine-tuning (e.g., hierarchical extractive encoders [23, 49]) or use long-input encoder–decoder generation [18, 50, 47], where factual accuracy requires attention [48]. In contrast, our method targets an extractive setting without fine-tuning on PubMed/ArXiv: it builds token-bounded context groups using semantic continuity and discourse cues, and performs candidate-guided rank-

ing within each context-consistent group, emphasizing contiguous context when selecting sentences for long scientific documents (Table 5). Unlike prior segment-based or hierarchical extractors, the ranking signal in our pipeline is derived from generated candidates that act as global salience proxies but are applied *within* context-consistent, token-bounded groups, without target-dataset fine-tuning.

Table 1: Summary of design characteristics in representative long-document summarization approaches. FT= fine-tuning on the target dataset. Context strategy: Semantic+disc.= semantic similarity with discourse markers (proposed method, Section 3); other strategies follow standard definitions.

Method	Type	FT	Context strategy
Top-Down [23]	Extr.	Yes	Hierarchical
Pegasus [18]	Abst.	Yes	Seq2seq
LED [50]	Abst.	Yes	Long-range attn.
BigBird [48]	Extr.	Yes	Sparse attn.
HiStruct+ [49]	Extr.	Yes	Hierarchical
Dancer [47]	Extr.	Varies	Divide-conquer
Ours	**Extr.**	**No**	**Semantic+disc.**

Algorithm 1 Compact overview of the proposed context-aware extractive pipeline.

Require: Sentences $(s_i)_{i=1}^{N}$; θ; budget L; generator T5; similarity Sim; k, K; n-gram blocks n_g, n_s; summary budget B.
Ensure: Summary S
$e_i \leftarrow \mathrm{SB}(s_i)$ for all i; $G \leftarrow [\,]$; $g \leftarrow [s_1]$
for $i = 2$ to N **do**
  if $\big(\cos(e_{i-1}, e_i) > \theta \vee \mathrm{Disc}(s_i)\big) \wedge (\mathrm{tokenCount}(g) + \mathrm{tokenCount}(s_i) \leq L)$ **then**
    append s_i to g
  else
    append g to G; $g \leftarrow [s_i]$
  end if
end for
append g to G
$T \leftarrow \mathrm{TopK}\big(\mathrm{DedupNgram}(\cup_{g \in G} \mathrm{T5}(g, k), n_g), K\big)$
$r_i \leftarrow \mathrm{Sim}(s_i, T)$ for all i
$S \leftarrow \mathrm{SelectTop}(s_{1:N}, r_{1:N}, B, n_s)$ ▷ keep original order
return S

3 Proposed method

We propose a context-aware extractive summarization model. The model has two components: a Context Detector (CD) and a Sentence Selector (SS). CD groups consecutive sentences into contextually coherent groups. SS generates multiple candidate texts per group using a pretrained T5 model. SS then ranks source sentences by semantic similarity to those candidates to produce an extractive summary. The SS builds upon the candidate-generation and ranking framework of Bishop et al. [41]. We adapt their approach to operate over context groups. The summarization pipeline is unsupervised with respect to the target datasets and does not require task-specific fine-tuning on ArXiv or PubMed; all neural components are used in a zero-shot manner on external scientific corpora, with no parameter updates.

Algorithm 1 summarizes the end-to-end procedure of our context-aware extractive pipeline, including the context grouping rule (cosine threshold and discourse-marker triggers), the token-budget constraint, candidate pooling with n-gram de-duplication, and the final sentence selection with redundancy control.

The subsequent subsections detail each stage in Algorithm 1 and report the parameter choices used in our experiments.

Here, "zero-shot" means *no training or fine-tuning on the target datasets (PubMed/ArXiv)*; we use pretrained models (e.g., Sentence-BERT and a T5 variant previously fine-tuned on S2ORC) *as-is*, without updating their parameters.

In our implementation, CD uses Sentence-BERT embeddings (the `allenai-specter` variant). Candidate generation uses a T5 model fine-tuned on S2ORC. We use frequency-based aggregation (optionally) and apply n-gram blocking to reduce redundancy. Sentence scoring supports several similarity backbones (BERTScore, SimCSE, and Sentence-BERT). The final summary is purely extractive: all output sentences are taken verbatim from the source document, and T5-generated texts serve only as intermediate candidates for similarity-based scoring. The overall architecture is illustrated in Figure 1.

Given a document D, we split it into sentences and apply the Context Detector (CD) to form budget-bounded context groups $G = \{g_1, \ldots, g_m\}$. We use *context group* (or simply *group*) to denote the contiguous, budget-bounded unit produced by the Context Detector. For each group g_j, the Sentence Selector generates k candidate texts T_j using a T5-based generator, pools and de-duplicates candidates via n-gram blocking (optionally frequency-weighted), and scores each sentence by similarity to the retained candidates. The top sentences are selected under a length constraint and output in original document order.

We first preprocess each document by splitting it into sentences $S = (s_1, s_2, \ldots, s_n)$. In the experiments, we use off-the-shelf sentence tokenizers (spaCy or NLTK) to obtain sentence boundaries. The implementation accepts either a raw string (segmented with spaCy/NLTK) or a pre-split list of sentences. Each sentence is treated as a string unit for embedding and similarity computations. To enforce transformer input-size constraints, we approximate token limits by cumulative token counts: $\mathrm{TokenEstimate}(g) \approx \sum_{s \in g} |s|_{\mathrm{tokens}}$. The code enforces that the cumulative token-count per group does not exceed 512 (token-based approximation). The CD grouping is formalized below.

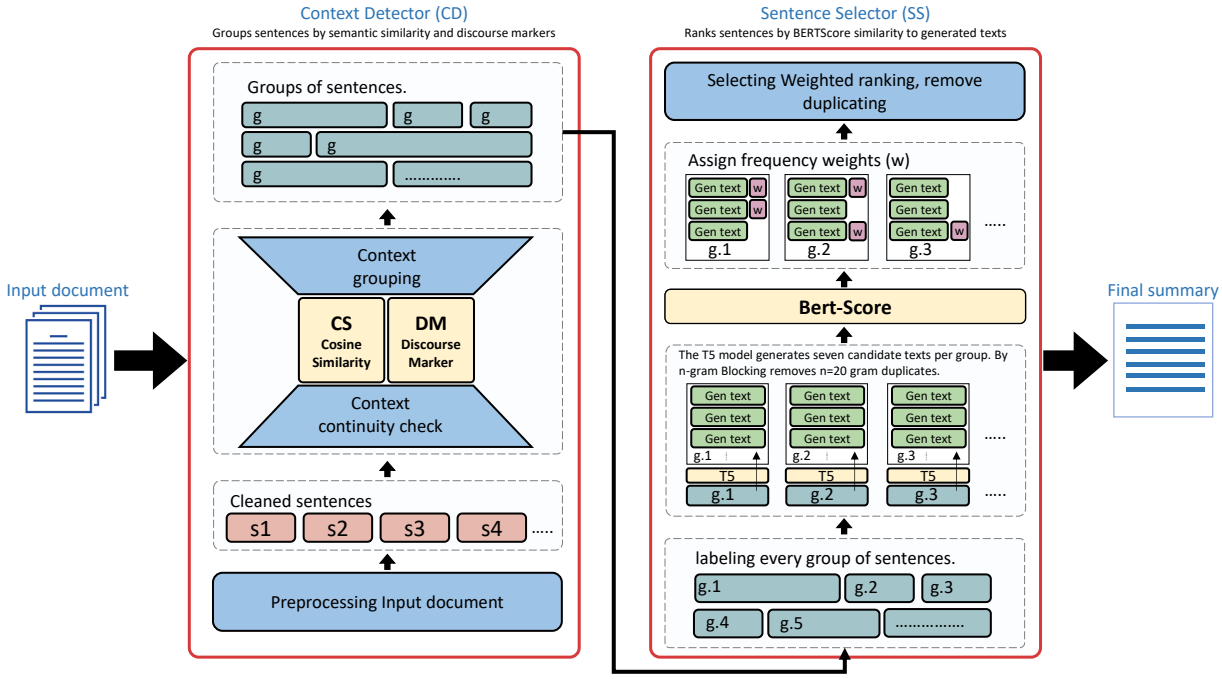

Figure 1: Overview of the proposed method, consisting of a Context Detector (CD) and a Sentence Selector (SS). CD groups sentences into contexts using discourse markers (DM) and cosine similarity (CS). SS generates candidates via a T5-based generator, applies n-gram blocking, and selects final summary sentences using a similarity-based ranking (BERTScore by default; SimCSE and Sentence-BERT are also supported).

$$g_j = \begin{cases} \{s_1\}, & \text{if } j = 1, \\ g_{j-1} \cup \{s_i\}, & \text{if DS}(s_{i-1}, s_i) \text{ and} \\ & \text{tokenCount}\Big(g_{j-1} \cup \{s_i\}\Big) \\ & \leq 512, \\ \{s_i\}, & \text{otherwise.} \end{cases} \quad (1)$$

Here g_j is the j-th contextual group formed from contiguous sentences. The numeric limit of 512 indicates an approximate token count that reflects the transformer's input.

The discourse markers are a curated list of words and short phrases that often indicate continuation or transition (we use a list derived and adapted from prior work). We define the predicate $DM(s_i)$ to be true when the *current* sentence s_i begins with a discourse marker from the list. The function $CS(s_{i-1}, s_i)$ computes the cosine similarity between Sentence-BERT embeddings of s_{i-1} and s_i (we use the `allenai-specter` model).

$$\text{DS}(s_{i-1}, s_i) \;=\; DM(s_i) \;\vee\; CS(s_{i-1}, s_i) > \theta \quad (2)$$

Here DS is a Boolean predicate that becomes true if the current sentence begins with a discourse marker or if the consecutive-sentence cosine similarity exceeds θ. In our experiments, we set $\theta = 0.4$ based on the sensitivity analysis in Section 4.2.4, which shows this value provides a consistent trade-off between coherent grouping and summarization performance

We merge s_i into the current context group if $DS(s_{i-1}, s_i)$ holds and the group budget is not exceeded; $DM(s_i)$ is computed using a curated discourse-marker list [42], and $CS(s_{i-1}, s_i)$ is the cosine similarity between consecutive sentence embeddings.

The overall workflow of the Context Detector (CD) component is illustrated in Algorithm 2.

Then the Sentence Selector (SS) uses the context groups produced by CD to find the most important sentences for the final extractive summary. The SS follows an unsupervised, candidate-generation-and-ranking approach. For candidate generation, we use a T5-based model [43] to produce multiple outputs per context group. Formally, for each context g_j we generate

$$\{t_{j,1}, t_{j,2}, \ldots, t_{j,k}\} = \text{t5\_gen}(g_j), \quad (3)$$

where $t_{j,\ell}$ is the ℓ-th text generated from context g_j.

After generation, texts from all groups are aggregated into a pool $T = \{t_{1,1}, \ldots, t_{1,k}, t_{2,1}, \ldots, t_{m,k}\}$, where m is the number of context groups and k is the number of texts per group. To improve diversity and reduce redundancy, we apply n-gram blocking (with $n = 20$ for candidate deduplication) and form a filtered pool $T^{(n)} \subseteq T$.

Algorithm 2 Context Detector (CD: Context grouping using DM and CS)

Input: Sentence list S
Output: Ordered context groups G
$n \leftarrow |S|$ ▷ number of sentences
Initialize $G \leftarrow []$
if $n = 0$ **then**
  return G
end if
Initialize current_group $\leftarrow [s_1]$
$\theta \leftarrow 0.4$
$M \leftarrow 512$ ▷ token-count approximation
for $i = 2$ **to** n **do**
  DS $\leftarrow DM(s_i) \vee CS(s_{i-1}, s_i) > \theta$
  if tokenCount(current_group) + tokenCount(s_i) > M **then**
    Append current_group to G
    current_group $\leftarrow [s_i]$
  end if
  if DS **and** tokenCount(current_group) + tokenCount(s_i) $\leq M$ **then**
    Append s_i to current_group
  else
    Append current_group to G
    current_group $\leftarrow [s_i]$
  end if
end for
if current_group not empty **then**
  Append current_group to G
end if
return G

Candidates in $T^{(n)}$ can be assigned frequency-based weights according to how often the same (or near-identical) candidate was generated across groups. In other words, aggregation supports optional frequency weighting (counts) rather than only equal weighting.

We compute similarity between each original sentence and the retained candidates using BERTScore [44] (precision/recall/F1). In the implementation, we use the F1 score as the candidate-to-sentence similarity measure; alternative similarity backbones (SimCSE or Sentence-BERT) are also supported.

Let $T^{(n)} = \{t_1, \ldots, t_K\}$ be the retained candidates, and let w_j denote the (optional) frequency weight of candidate t_j. We compute a final score d_i for each sentence s_i as follows:

$$d_i \;=\; \sum_{j=1}^{K} w_j \;\times\; \mathrm{BS}_{\mathrm{F1}}(s_i, t_j). \tag{4}$$

Here $\mathrm{BS}_{\mathrm{F1}}(s_i, t_j)$ denotes the BERTScore F1 between sentence s_i and candidate t_j. When Sentence-BERT is used as the similarity backbone, the implementation computes cosine distance between vectors (and the ranking logic handles this by minimizing distance rather than maximizing similarity). Sentences are ranked by d_i (descending) and the top-ranked sentences are selected (with n-gram blocking applied at selection time) and returned in original document order. The selection procedure is described in Algorithm 3.

Algorithm 3 Sentence Selector

Input: Context groups $G = \{g_1, \ldots, g_m\}$, Document sentences S
Output: Selected sentences (summary) S'
$k \leftarrow$ number of texts to generate per group (default: 7)
$target\_sentences \leftarrow 9$
Initialize candidate pool $T \leftarrow []$, weights $W \leftarrow []$, scores $D \leftarrow []$
for each context g_j in G **do**
  Generate $\{t_{j,1}, \ldots, t_{j,k}\} \leftarrow$ t5_gen(g_j)
  $T_j^{(n)} \leftarrow$ N-gram_Blocking($\{t_{j,\ell}\}, n = 20$)
  Add elements of $T_j^{(n)}$ to pool T
end for
Compute frequency weights W for each unique candidate in T (optionally)
for each candidate t in T **do**
  Compute BERTScore F1 between t and each sentence $s_i \in S$
  Accumulate weighted scores d_i += $w_t \times \mathrm{BS}_{\mathrm{F1}}(s_i, t)$
end for
$sorted\_indices \leftarrow$ indices of sentences s_i sorted by d_i (descending)
$selected\_sentences \leftarrow []$
for idx in $sorted\_indices$ **do**
  candidate $\leftarrow S$[idx]
  if N-gram_Block_Check(candidate, selected_sentences, n=10) = False **then**
    Append candidate to $selected\_sentences$
    if $|selected\_sentences| = target\_sentences$ **then**
      break
    end if
  end if
end for
Sort $selected\_sentences$ by document order
$S' \leftarrow$ selected_sentences
return S'

4 Experiment

4.1 Datasets

We have selected two popular datasets from the long document datasets: one from the scientific field and the other from the medical field. The data are obtained from [45] and are available on huggingface.co. We extract the articles from arXiv.org and PubMed.com. Every single instance in the dataset is a scientific paper. The article abstracts serve as the ground truth. The datasets are divided into training,

validation, and testing sets in specific proportions, which we also apply in our work. The statistics of these sets are presented in Table 2. Our primary focus is on the test set, as our approach requires no training on the target datasets.

Table 2: Number of document–abstract pairs in the training, validation, and test splits for the ArXiv and PubMed long-document benchmarks. Since our method is unsupervised, evaluation is performed exclusively on the test split.

Datasets	Train Set	Val Set	Test Set
ArXiv	203,037	6,436	6,440
PubMed	119,224	6,633	6,658

4.2 Experimental settings

4.2.1 Preprocessing

To avoid evaluation leakage, the reference abstract is used only as the gold summary and is excluded from the input. If an "Abstract" section is present in the full text, it is removed prior to processing. We normalize whitespace and split each input document into sentences using NLTK for sentence segmentation and tokenization; token length is used as an approximate budget for later stages.

For evaluation, the model input consists only of the document body (article text); the reference abstract is used exclusively as the gold summary for computing ROUGE/BERTScore and is never included in the input. We remove empty or malformed sentences after sentence splitting and preserve the original sentence order for grouping and selection. Sentence segmentation errors are handled conservatively: empty or malformed segments are discarded, and documents with fewer than two valid sentences after filtering are returned unchanged. All subsequent stages operate on the resulting sentence list, and no additional sentence re-ordering is performed.

4.2.2 Computational resources

All experiments were run using PyTorch with CUDA acceleration on a single NVIDIA A100 GPU (40 GB VRAM). Since our pipeline performs no supervised fine-tuning on the target datasets (PubMed/ArXiv), a single-GPU configuration is sufficient for the full evaluation. End-to-end processing time averages 40 s per document, including context grouping, candidate generation, BERTScore-based ranking and sentence selection; this cost is expected, given the candidate generation step, and optimization of inference speed is left for future work.

4.2.3 Model configuration

Our model combines two neural components that work together during summarization. For salient text generation, we use the T5-base model fine-tuned on the Semantic Scholar Open Research Corpus (S2ORC). The version used is `doc2query/S2ORC-t5-base-v1`, which is designed to create query-like representations of scientific text. Each generated text contains up to 64 tokens. We apply top-k sampling $k = 10$ to maintain a balance between diversity and coherence, and set the temperature to 0.5 to control randomness in generation.

The model generates seven candidates for each section of a document. For all sections, we keep the top 25 candidates for the next step.

We performed a lightweight sensitivity analysis on the candidate-generation settings using 100 ArXiv test documents. We varied the number of candidates per section ($k \in \{3, 5, 7\}$) and the number of retained texts ($K \in \{15, 25, 35\}$), keeping other parameters fixed (top-k sampling $= 10$, temperature $= 0.5$, max length $= 64$). Table 4 reports ROUGE scores; the default setting ($k = 7, K = 25$) provided the best trade-off in our experiments.

To calculate sentence similarity and perform ranking, we test three models: BERTScore, using the bert-base-uncased model with 12 layers and a batch size of 64; SimCSE, which measures contextual similarity; and Sentence-BERT, using the allenai-specter variant trained on scientific papers. The main configuration uses BERTScore because its similarity scores match human judgments in extractive summarization.

All neural components are obtained from the Hugging Face Hub using fixed version identifiers to maintain consistent weights and tokenization. We report the exact Hugging Face model identifiers for all pretrained components so that the same checkpoints can be retrieved and evaluated under identical preprocessing and hyperparameters.

4.2.4 Hyperparameters

The Context Detector groups sentences by combining semantic similarity and discourse structure. Sentences are clustered into coherent groups using a cosine similarity threshold of 0.4. Sentence embeddings are generated with the `allenai-specter` model from the Sentence-BERT family. The threshold was selected via an ablation study on 100 documents from the arXiv test set, testing values ranging from 0.2 to 0.7, as shown in Table 3. Thresholds below 0.3 produced overly large clusters that merged unrelated passages. These clusters frequently exceeded the 512-token limit, requiring splits that reduced coherence. In implementation, we approximate the Transformer budget using word-based token counts; this can be replaced with tokenizer-based token counts for exact budgeting. Thresholds above 0.6 produced excessively small groups, with individual sentences frequently treated as isolated contexts. A threshold of 0.4 struck the best balance, yielding an average of 12.3 context groups per document, with an average of 3.8 sentences per context group.

Using the same 100 ArXiv documents, we vary the candidate-generation configuration by changing the number of generated texts per section (k) and the number of retained candidates (K), while keeping other decoding pa-

Table 3: Impact of cosine similarity threshold on context grouping and summarization performance (evaluated on a randomly sampled held-out subset of 100 ArXiv documents). Bold values represent the highest scores.

Threshold	Avg. Groups	Avg. Group Size	R-1	R-2	R-L
0.2	6.2	7.1	50.8	19.1	45.7
0.3	8.7	5.3	52.3	19.8	47.1
0.4	**12.3**	**3.8**	**52.9**	**20.1**	**47.6**
0.5	16.8	2.6	52.1	19.5	46.9
0.6	24.1	1.8	50.6	18.7	45.8
0.7	31.4	1.3	48.9	17.9	44.3

rameters fixed (top-k sampling $= 10$, temperature $= 0.5$, max length $= 64$). As shown in Table 4, increasing k from 3 to 7 yields consistent ROUGE improvements, and the default setting ($k = 7$, $K = 25$) achieves the best ROUGE-1/ROUGE-2/ROUGE-L among the tested configurations.

Table 4: Sensitivity analysis of candidate-generation settings on 100 ArXiv documents (other parameters fixed; ROUGE-1/2/L reported).

Setting	R-1	R-2	R-L
$k = 3$, $K = 25$	51.8	19.4	46.8
$k = 5$, $K = 25$	52.5	19.8	47.3
$k = 7$, $K = 25$ (default)	52.9	20.1	47.6
$k = 7$, $K = 15$	52.3	19.7	47.0
$k = 7$, $K = 35$	52.6	19.9	47.4

The 512-token cap is enforced to avoid truncation under common Transformer input constraints, using the same token-count approximation described in Section 4.2.4. We apply n-gram blocking (20-gram for candidate generation and 10-gram for sentence selection) to control redundancy; in our ArXiv-100 sensitivity checks, moderate blocking values preserved ROUGE trends while reducing repetitive sentence inclusion.

This configuration provided the highest ROUGE scores within the sampled subset. Context detection also relies on explicit discourse markers in addition to similarity scores. The system recognizes 24 markers, such as ”moreover,” ”furthermore,” ”in addition,” ”additionally,” ”consequently,” ”for instance,” and ”indeed.” A sentence is merged into the current group if $DS(s_{i-1}, s_i)$ holds and adding it does not exceed the group budget (token-count approximation): $\text{tokenCount}(g_{j-1} \cup \{s_i\}) \leq 512$; otherwise, a new group is started (Eq. (1)–(2)). This mixed rule ensures both semantic and structural coherence in grouping.

The n-gram blocking is performed in 2 stages to reduce redundancy; 20-gram blocking eliminates redundancy or near-redundancy among sequences sharing 20 consecutive words during salient text generation. This process ensures that the candidate's textual outputs are not similar. 10-gram blocking prevents redundant material by allowing only sentences with identical 10-word spans. The length of the summary is controlled by aiming at nine sentences or about 250 tokens and a 50-token margin to provide a natural end to sentences.

Each sentence receives a score indicating its importance to the document. The score is based on its similarity to all retained salient texts, weighted by how frequently those appear in different document sections. This weighting highlights sentences that align with the most consistently generated salient content, improving both coverage and balance in the final summary.

4.3 Evaluation metrics

We evaluate generated summaries using ROUGE [46], reporting ROUGE-1, ROUGE-2, and ROUGE-L (LCS) against the reference abstracts to enable direct comparison with prior work on the ArXiv and PubMed long-document benchmarks. While ROUGE mainly reflects lexical overlap and may not fully capture semantic adequacy, it remains the standard automatic metric for long-document summarization.

To complement ROUGE with a semantic measure, we additionally report BERTScore-F1 [44]. We also perform a paired t-test on per-document scores between our method and the strongest baseline and report significance at the 95% confidence level.

5 Main results and discussion

5.1 Results

Table 5 presents the main results of our model across both datasets. The table reports ROUGE-1, ROUGE-2, and ROUGE-L on PubMed and ArXiv.

We compare against representative long-document baselines, including Top-Down Transformer [23], Pegasus [18], Dancer [47], BigBird [48], HiStruct+ [49], and LED [50].

We additionally report BERTScore-F1 (Table 6) to assess semantic similarity beyond lexical overlap; the overall trend is consistent with Table 5. We further assess statistical significance using paired t-tests on per-document scores relative to the strongest baseline (Top-Down Transformer) (Table 7). Our method achieves statistically significant improvements in ROUGE-1, ROUGE-L, and BERTScore-F1 on both datasets at the 95% confidence level. On PubMed, all metrics, including ROUGE-2, show significant gains. On ArXiv, ROUGE-2 shows a non-significant decrease

Table 5: A comparison of performance between the proposed model and baseline models on the PubMed and ArXiv long-document datasets. The values reported are ROUGE-1 (R-1), ROUGE-2 (R-2), and ROUGE-L (R-L). Bold values indicate the best scores per metric.

	PubMed			ArXiv		
	R-1	R-2	R-L	R-1	R-2	R-L
Top-Down Transformer	51.05	23.26	46.47	50.95	**21.93**	45.61
Pegasus	47.06	19.81	41.11	44.21	16.95	38.83
Dancer	46.34	19.97	42.42	45.01	17.60	40.56
BigBird	46.32	20.65	42.33	46.63	19.02	41.77
HiStruct+	46.03	19.12	41.95	45.01	17.82	39.83
LED	46.38	20.17	41.92	45.17	17.61	40.10
Our Model	**53.0**	**24.32**	**47.7**	**53.7**	20.40	**48.4**

Table 6: Semantic evaluation using BERTScore-F1 (higher is better; reported as $\times 100$) for the strongest representative baselines in Table 5.

Model	PubMed	ArXiv
Top-Down Transformer	87.45	87.82
HiStruct+	85.83	85.41
LED	86.02	85.68
Our Model	**88.71**	**88.93**

($\Delta = -1.53, p = 0.125$), consistent with the use of n-gram blocking, which reduces redundancy but can also reduce bigram overlap with the reference, as discussed below.

Table 7: Paired t-test results comparing our method with Top-Down Transformer on each dataset (per-document scores). "Sig." indicates $p < 0.05$.

Metric	Mean Δ	p-value	Sig.
PubMed dataset			
ROUGE-1	+1.95	0.0021	Yes
ROUGE-2	+1.06	0.0389	Yes
ROUGE-L	+1.23	0.0142	Yes
BERTScore-F1	+1.26	0.0089	Yes
ArXiv dataset			
ROUGE-1	+2.75	0.0003	Yes
ROUGE-2	-1.53	0.1247	No
ROUGE-L	+2.79	0.0005	Yes
BERTScore-F1	+1.11	0.0198	Yes

5.2 Discussion

Across both datasets, our approach achieves the highest scores on PubMed (ROUGE-1: 53.0, ROUGE-2: 24.32, ROUGE-L: 47.7) among all compared models, and achieves the highest ROUGE-1 (53.7) and ROUGE-L (48.4) on ArXiv; however, the Top-Down Transformer attains the best ArXiv ROUGE-2 score (21.93). These results suggest that context-aware grouping contributes to sentence selection and ordering. The ROUGE-1 improvement suggests that the context-aware selection tends to identify sentences containing salient terms. Furthermore, the improved ROUGE-L suggests better sequence-level overlap with the reference abstracts.

From a mechanism perspective, the gains come from aligning sentence ranking with document-level structure: grouping sentences into budget-bounded, context-consistent groups reduces cross-topic interference during candidate generation and scoring, and encourages selection from coherent regions that reflect the paper's main contributions and findings. This is reflected in ROUGE-1/ROUGE-L and BERTScore gains, which emphasize salient content and sequence-level overlap rather than exact bigram matching.

Table 1 contrasts representative long-document strategies. In our unified evaluation (Table 5), fine-tuned hierarchical extractors (e.g., Top-Down/HiStruct+) are strong baselines, while long-input encoder–decoder models (Pegasus/LED) rely on substantial pretraining and fine-tuning, and may be sensitive to factuality constraints in scientific writing. Our method targets a different setting by enforcing explicit context continuity via token-bounded semantic groups and applying candidate-guided ranking within each group without PubMed/ArXiv fine-tuning.

On ArXiv, Top-Down Transformer achieves the highest ROUGE-2 (21.93), while our method obtains 20.40. Statistical analysis confirms this 1.53-point difference is not significant ($p = 0.125$, Table 7) and is consistent with our design choice: the n-gram blocking mechanism prioritizes summary diversity and semantic similarity (supported by higher ROUGE-1, ROUGE-L, and BERTScore on both datasets) over exhaustive bigram matching. We use 10-gram blocking during sentence selection to reduce redundancy; however, it can also lower ROUGE-2 by filtering sentences that contain frequently recurring bigrams common in scientific writing.

The mechanism is effective at avoiding word-for-word replication of long sentences, but it can also filter sentences that contain recurring yet informative bigrams, including

domain-specific terminology that naturally appears in scientific abstracts. For example, phrases such as ”we demonstrate that” and collocations such as ”neural network architecture” frequently appear in both source documents and reference abstracts. This trade-off is particularly evident on ArXiv, where repetitive technical terminology is more common. As an extractive method, our summaries preserve source sentences, which reduces the risk of introducing unsupported content.

The performance pattern across different ROUGE metrics reveals fundamental characteristics of our extractive approach. The context detection module, which groups sentences based on a 0.4 cosine similarity threshold and discourse markers, excels at identifying semantically coherent groups that capture the document's main themes, translating directly to high ROUGE-1 scores. Table 3 further serves as an implicit ablation of this module: degraded grouping configurations consistently reduce ROUGE across all metrics, confirming the contribution of context-aware boundaries to overall performance.

While ROUGE-2 is more sensitive to redundancy control, our approach selects complete sentences from the source text and preserves their original wording. The n-gram blocking constraints help prevent redundancy that can otherwise arise in extractive summaries of long documents, where similar concepts are elaborated across sections. Our method requires no fine-tuning on PubMed/ArXiv and may generalize across domains.

This setting evaluates generalization without *target-dataset* supervision, but it does not claim that the underlying pretrained components were trained without supervision on external scientific corpora. The comparable performance on both PubMed and ArXiv datasets suggests that the approach is potentially applicable to diverse scientific domains. This is because the basic ideas behind context detection via semantic similarity and discourse markers, query generation via fine-tuned T5, and weighted sentence ranking via BERTScore reflect common aspects of scientific document structure rather than patterns specific to one field.

These results advance the field by showing that explicit context continuity within token budgets can reduce reliance on target-dataset fine-tuning for scientific document summarization. Our context-consistent grouping combined with candidate-guided ranking injects document-level salience while preserving extractive faithfulness, enabling competitive cross-domain performance without labeled supervision on the target datasets. This provides a practical path toward deployable systems in resource-constrained settings.

Our approach has limitations that follow directly from its design choices. First, when sentence boundaries are noisy (e.g., in poorly formatted PDFs) or discourse markers are sparse or non-standard, context grouping may be less reliable and could merge or split topics inappropriately. Second, in documents with heavy repetition of domain-specific bigrams or formulaic phrasing, n-gram blocking can remove sentences that contain informative recurring terminology, which may reduce ROUGE-2 as observed on ArXiv. Third, if the generated candidates are low-quality (e.g., for highly technical sections with limited natural-language cues), ranking can overweight less informative sentences. These cases suggest that the method is best suited to long scientific articles with reasonably well-formed sentence grouping and standard discourse structure.

6 Conclusion

We have presented a context-aware extractive summarization pipeline for long scientific documents that forms budget-bounded context groups and ranks sentences using generated candidates and semantic similarity scoring. Experiments on PubMed and ArXiv show consistent gains in ROUGE-1/ROUGE-L and improved semantic similarity (BERTScore).

Limitations include sensitivity to noisy sentence boundaries and non-standard discourse structure, potential suppression of informative recurring terminology due to n-gram blocking, and dependence on candidate quality in highly technical sections. Future work will investigate more robust grouping signals for extremely long documents, adaptive redundancy control that preserves domain-specific terminology, and improved candidate generation, together with broader evaluation and targeted qualitative analysis.

References

[1] Sheher Bano, Shah Khalid (2022) BERT-based extractive text summarization of scholarly articles: A novel architecture, *2022 International Conference on Artificial Intelligence of Things (ICAIoT)*, IEEE, pp. 1–5, doi: `https://doi.org/10.1109/ICAIoT57170.2022.10121826`.

[2] Inderjeet Mani, Mark T. Maybury (1999) Advances in Automatic Text Summarization, , The MIT Press, ISBN: 9780262133593.

[3] Nouf Ibrahim Altmami, Mohamed El Bachir Menai (2022) Automatic summarization of scientific articles: A survey, *Journal of King Saud University-Computer and Information Sciences*, 34(4), pp. 1011–1028, Elsevier, doi: `https://doi.org/10.1016/j.jksuci.2020.04.020`.

[4] Minakshi Tomer, Manoj Kumar (2022) Multi-document extractive text summarization based on firefly algorithm, *Journal of King Saud University-Computer and Information Sciences*, 34(8), pp. 6057–6065, Elsevier, doi: `https://doi.org/10.1016/j.jksuci.2021.04.004`.

[5] Adhika Pramita Widyassari, Supriadi Rustad, Guruh Fajar Shidik, Edi Noersasongko, Abdul Syukur, Affandy Affandy, et al. (2022) Review of automatic text summarization techniques & methods, *Journal of King Saud University-Computer and Information Sciences*, 34(4), pp. 1029–1046, Elsevier, doi: https://doi.org/10.1016/j.jksuci.2020.05.006.

[6] Byron C. Wallace, Sayantan Saha, Frank Soboczenski, Iain J. Marshall (2021) Generating (factual?) narrative summaries of RCTs: Experiments with neural multi-document summarization, *AMIA Summits on Translational Science Proceedings*, 2021, pp. 605, American Medical Informatics Association, doi: https://doi.org/10.48550/arXiv.2008.11293.

[7] Sheher Bano, Shah Khalid, Nasser Mansoor Tairan, Habib Shah, Hasan Ali Khattak (2023) Summarization of scholarly articles using BERT and BiGRU: Deep learning-based extractive approach, *Journal of King Saud University-Computer and Information Sciences*, 35(9), 101739, Elsevier, doi: https://doi.org/10.1016/j.jksuci.2023.101739.

[8] Dandan Huang, Leyang Cui, Sen Yang, Guangsheng Bao, Kun Wang, Jun Xie, Yue Zhang (2020) What have we achieved on text summarization?, *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Association for Computational Linguistics, pp. 446–469, doi: https://doi.org/10.18653/v1/2020.emnlp-main.33.

[9] Hans Peter Luhn (1957) A statistical approach to mechanized encoding and searching of literary information, *IBM Journal of Research and Development*, 1(4), pp. 309–317, doi: https://doi.org/10.1147/rd.14.0309.

[10] Karen Sparck Jones (1972) A statistical interpretation of term specificity and its application in retrieval, *Journal of Documentation*, 28(1), pp. 11–21, doi: https://doi.org/10.1108/eb026526.

[11] Rada Mihalcea, Paul Tarau (2004) Textrank: Bringing order into text, *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, pp. 404–411.

[12] Jacob Devlin, Ming-Wei Chang, Kenton Lee, Kristina Toutanova (2019) BERT: Pre-training of deep bidirectional transformers for language understanding, *Proceedings of NAACL-HLT 2019 (Long and Short Papers)*, pp. 4171–4186, doi: https://doi.org/10.18653/v1/N19-1423.

[13] Ming Ding, Chang Zhou, Hongxia Yang, Jie Tang (2020) Cogltx: Applying BERT to long texts, *Advances in Neural Information Processing Systems (NeurIPS)*, 33, pp. 12792–12804.

[14] Arash Afkanpour, Shabir Adeel, Hansenclever Bassani, Arkady Epshteyn, Hongbo Fan, Isaac Jones, Mahan Malihi, Adrian Nauth, Raj Sinha, Sanjana Woonna, Shiva Zamani, Elli Kanal, Mikhail Fomitchev, Donny Cheung (2022) BERT for Long Documents: A Case Study of Automated ICD Coding, *Proc. LOUHI 2022*, pp. 100–107, doi: https://doi.org/10.48550/arXiv.2211.02519.

[15] Yang Liu, Mirella Lapata (2019) Text Summarization with Pretrained Encoders, *Proc. EMNLP-IJCNLP 2019*, pp. 3730–3740, doi: https://doi.org/10.18653/v1/D19-1387.

[16] Isabel Cachola, Kyle Lo, Arman Cohan, Daniel Weld (2020) TLDR: Extreme Summarization of Scientific Documents, *Findings of the Association for Computational Linguistics: EMNLP 2020*, Association for Computational Linguistics, pp. 4766–4777, doi: https://doi.org/10.18653/v1/2020.findings-emnlp.428.

[17] Shusheng Xu, Xingxing Zhang, Yi Wu, Furu Wei, Ming Zhou (2020) Unsupervised Extractive Summarization by Pre-training Hierarchical Transformers, *Findings of the Association for Computational Linguistics: EMNLP 2020*, doi: https://doi.org/10.18653/v1/2020.findings-emnlp.161.

[18] Jingqing Zhang, Yao Zhao, Mohammad Saleh, Peter Liu (2020) Pegasus: Pre-training with extracted gap-sentences for abstractive summarization, *International Conference on Machine Learning*, PMLR, pp. 11328–11339, doi: https://doi.org/10.48550/arXiv.1912.08777.

[19] Ming Zhong, Pengfei Liu, Yiran Chen, Danqing Wang, Xipeng Qiu, Xuanjing Huang (2020) Extractive Summarization as Text Matching, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, Association for Computational Linguistics, pp. 6197–6208, doi: https://doi.org/10.18653/v1/2020.acl-main.552.

[20] Zi-Yi Dou, Pengfei Liu, Hiroaki Hayashi, Zhengbao Jiang, Graham Neubig (2021) GSum: A general framework for guided neural abstractive summarization, *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 4830–4842, doi: https://doi.org/10.18653/v1/2021.naacl-main.384.

[21] Zhiguo Wang, Patrick Ng, Xiaofei Ma, Ramesh Nallapati, Bing Xiang (2019) Multi-passage BERT: A Globally Normalized BERT Model for Open-domain Question Answering, *Proc. EMNLP-IJCNLP 2019*, pp. 5878–5882, doi: https://doi.org/10.18653/v1/D19-1599.

[22] Abigail See, Peter J. Liu, Christopher D. Manning (2017) Get to the point: Summarization with pointer-generator networks, *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1073–1083, doi: https://doi.org/10.18653/v1/P17-1099.

[23] Bo Pang, Erik Nijkamp, Wojciech Kryściński, Silvio Savarese, Yingbo Zhou, Caiming Xiong (2023) Long document summarization with top-down and bottom-up inference, *Findings of the Association for Computational Linguistics: EACL 2023*, pp. 1267–1284, doi: https://doi.org/10.18653/v1/2023.findings-eacl.94.

[24] Majid Ramezani, Mohammad-Reza Feizi-Derakhshi (2016) Achieving More Coherent Summaries in Automatic Text Summarization; an Ontology-based Approach, *British Journal of Mathematics & Computer Science*, 19(6), pp. 1–15, doi: https://doi.org/10.9734/BJMCS/2016/27549.

[25] Hao Zheng, Mirella Lapata (2019) Sentence Centrality Revisited for Unsupervised Summarization, *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, Association for Computational Linguistics, pp. 6236–6247, doi: https://doi.org/10.18653/v1/P19-1628.

[26] Xinnian Liang, Shuangzhi Wu, Mu Li, Zhoujun Li (2021) Improving unsupervised extractive summarization with facet-aware modeling, *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pp. 1685–1697, doi: https://doi.org/10.18653/v1/2021.findings-acl.147.

[27] Ani Nenkova, Kathleen McKeown (2012) A survey of text summarization techniques, *Mining Text Data*, Springer, pp. 43–76, doi: https://doi.org/10.1016/j.nlp.2024.100070.

[28] Ramesh Nallapati, Bowen Zhou, Mingbo Ma (2016) Classify or select: Neural architectures for extractive document summarization, *arXiv preprint arXiv:1611.04244*, doi: https://doi.org/10.48550/arXiv.1611.04244.

[29] Qingyu Zhou, Nan Yang, Furu Wei, Shaohan Huang, Ming Zhou, Tiejun Zhao (2018) Neural document summarization by jointly learning to score and select sentences, *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Association for Computational Linguistics, pp. 654–663, doi: https://doi.org/10.18653/v1/P18-1061.

[30] Jingren Zhang, Fang'ai Liu, Weizhi Xu, Hui Yu (2019) Feature fusion text classification model combining CNN and BiGRU with multi-attention mechanism, *Future Internet*, 11(11), 237, MDPI, doi: https://doi.org/10.3390/fi11110237.

[31] Hamad Zogan, Imran Razzak, Shoaib Jameel, Guandong Xu (2021) DepressionNet: Learning Multimodalities with User Post Summarization for Depression Detection on Social Media, *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, Association for Computing Machinery, pp. 133–142, doi: https://doi.org/10.1145/3404835.3462938.

[32] Jianpeng Cheng, Mirella Lapata (2016) Neural Summarization by Extracting Sentences and Words, *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Association for Computational Linguistics, pp. 484–494, doi: https://doi.org/10.18653/v1/P16-1046.

[33] Aishwarya Jadhav, Vaibhav Rajan (2018) Extractive summarization with swap-net: Sentences and words from alternating pointer networks, *ACL 2018-56th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference (Long Papers)*, Association for Computational Linguistics, pp. 142–151, doi: https://doi.org/10.18653/v1/P18-1014.

[34] Keping Bi, Rahul Jha, Bruce Croft, Asli Celikyilmaz (2021) AREDSUM: Adaptive Redundancy-Aware Iterative Sentence Ranking for Extractive Document Summarization, *Proc. EACL 2021 (Main Volume)*, pp. 281–291, doi: https://doi.org/10.18653/v1/2021.eacl-main.22.

[35] Ting Wang, Chuan Yang, Maoyang Zou, Jiaying Liang, Dong Xiang, Wenjie Yang, Hongyang Wang, Jia Li (2024) A study of extractive summarization of long documents incorporating local topic and hierarchical information, *Scientific Reports*, 14(1), 10140, Nature Publishing Group UK London, doi: https://doi.org/10.1038/s41598-024-60779-z.

[36] Chunlong Han, Jianzhou Feng, Haotian Qi (2024) Topic model for long document extractive summarization with sentence-level features and dynamic memory unit, *Expert Systems with Applications*, 238, 121873, Elsevier, doi: https://doi.org/10.1016/j.eswa.2023.121873.

[37] Aytuğ Onan, Hesham A. Alhumyani (2024) FuzzyTP-BERT: Enhancing extractive text summarization with fuzzy topic modeling and transformer networks, *Journal of King Saud University – Computer and Information Sciences*, 36(6), 102080, Elsevier, doi: https://doi.org/10.1016/j.jksuci.2024.102080.

[38] Qianqian Xie, Jennifer A. Bishop, Prayag Tiwari, Sophia Ananiadou (2022) Pre-trained language models with domain knowledge for biomedical extractive summarization, *Knowledge-Based Systems*,

252, 109460, Elsevier, doi: https://doi.org/10.1016/j.knosys.2022.109460.

[39] Mahira Kirmani, Gagandeep Kour, Mudasir Mohd, Nasrullah Sheikh, Dawood Ashraf Khan, Zahid Maqbool, Mohsin Altaf Wani, Abid Hussain Wani (2024) Biomedical semantic text summarizer, *BMC Bioinformatics*, 25(1), 152, Springer, doi: https://doi.org/10.1186/s12859-024-05712-x.

[40] Xiaoyong Chen, Zhiqiang Chen, Shi Cheng (2025) CoTHSSum: Structured long-document summarization via chain-of-thought reasoning and hierarchical segmentation, *Journal of King Saud University Computer and Information Sciences*, 37, 40, Springer, doi: https://doi.org/10.1007/s44443-025-00041-2.

[41] Jennifer Bishop, Qianqian Xie, Sophia Ananiadou (2022) GenCompareSum: a hybrid unsupervised summarization method using salience, *Proceedings of the 21st Workshop on Biomedical Language Processing*, pp. 220–240, doi: https://doi.org/10.18653/v1/2022.bionlp-1.22.

[42] Mehrdad Vasheghani Farahani, Zahra Ghane (2022) Unpacking the function(s) of discourse markers in academic spoken English: a corpus-based study, *The Australian Journal of Language and Literacy*, 45(1), pp. 49–70, Springer, doi: https://doi.org/10.1007/s44020-022-00005-3.

[43] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, Peter J. Liu (2020) Exploring the limits of transfer learning with a unified text-to-text transformer, *Journal of Machine Learning Research*, 21(140), pp. 1–67.

[44] Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, Yoav Artzi (2019) BERTScore: Evaluating Text Generation with BERT, *International Conference on Learning Representations*, doi: https://doi.org/10.48550/arXiv.1904.09675.

[45] Arman Cohan, Franck Dernoncourt, Doo Soon Kim, Trung Bui, Seokhwan Kim, Walter Chang, Nazli Goharian (2018) A Discourse-Aware Attention Model for Abstractive Summarization of Long Documents, *Proc. NAACL-HLT 2018, Vol. 2 (Short Papers)*, pp. 615–621, doi: https://doi.org/10.18653/v1/N18-2097.

[46] Chin-Yew Lin (2004) ROUGE: A Package for Automatic Evaluation of Summaries, *Text Summarization Branches Out*, Association for Computational Linguistics, pp. 74–81.

[47] Alexios Gidiotis, Grigorios Tsoumakas (2020) A divide-and-conquer approach to the summarization of long documents, *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 28, pp. 3029–3040, IEEE, doi: https://doi.org/10.1109/TASLP.2020.3037401.

[48] Manzil Zaheer, Guru Guruganesh, Kumar Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, et al. (2020) Big Bird: Transformers for longer sequences, *Advances in Neural Information Processing Systems*, 33, pp. 17283–17297, doi: https://doi.org/10.48550/arXiv.2007.14062.

[49] Qianqian Ruan, Malte Ostendorff, Georg Rehm (2022) HiStruct+: Improving Extractive Text Summarization with Hierarchical Structure Information, *Findings of the Association for Computational Linguistics: ACL 2022*, Association for Computational Linguistics, pp. 1292–1308, doi: https://doi.org/10.18653/v1/2022.findings-acl.102.

[50] Iz Beltagy, Matthew E. Peters, Arman Cohan (2020) Longformer: The Long-Document Transformer, *arXiv preprint arXiv:2004.05150*, doi: https://doi.org/10.48550/arXiv.2004.05150.