电子科技大学
University of Electronic Science and Technology of China

TEXT SUMMARIZATION VIA GLOBAL STRUCTURE AWARENESS

2026.03.25

Jiaquan Zhang, Chaoning Zhang,Shuxu Chen,Yibei Liu, Chenghao Li,Qigan Sun,
Shuai Yuan,Fachrina Dewi Puspitasari, Dongshen Han,Guoqing Wang, , Sung-Ho Bae，Yang Yang

1 Introduction & Motivation

Computational Cost Challenged

- Long documents significantly increase processing cost in NLP tasks.
- LLMs suffer from context window limitations and high inference cost.
- Redundant information leads to inefficient computation and resource waste.
- Large-scale long-text processing limits practical deployment.

Compression & Performance Challenges

- Existing summarization relies on local sentence-level selection, lacking global understanding.
- Model-based methods improve accuracy but suffer from poor scalability.
- LLM-based summarization has high computational overhead.
- Summarization often leads to:
  Loss of logical coherence
  Broken reasoning chains
  Reduced downstream performance

1 Introduction & Motivation

Methods ⟷ Limitation

Sentence-level pruning	⟷	only captures local semantics
Neural models (BERTSum, etc.)	⟷	expensive for long texts
LLM-based methods	⟷	high cost, low efficiency

Core Question：

How to achieve efficient summarization while preserving global semantic structure and logical coherence?

1 Introduction & Motivation

Why Global Structure (TDA)?

- Text is not just sequential, but structurally organized in semantic space
- Need to capture:
  Semantic clusters (topics)
  Logical dependencies (cross-paragraph relations)

Key Advantages

- Global awareness: captures document-level structur
- Noise filtering: distinguishes core vs redundant content
- Structure preservation: maintains reasoning chains

2 Methodology

Framework Overview

Our method, GloSA-sum, is a global structure-aware summarization framework.
It first extracts the semantic and logical backbone of a document through a one-time topological analysis, and then performs iterative compression guided by this structure.

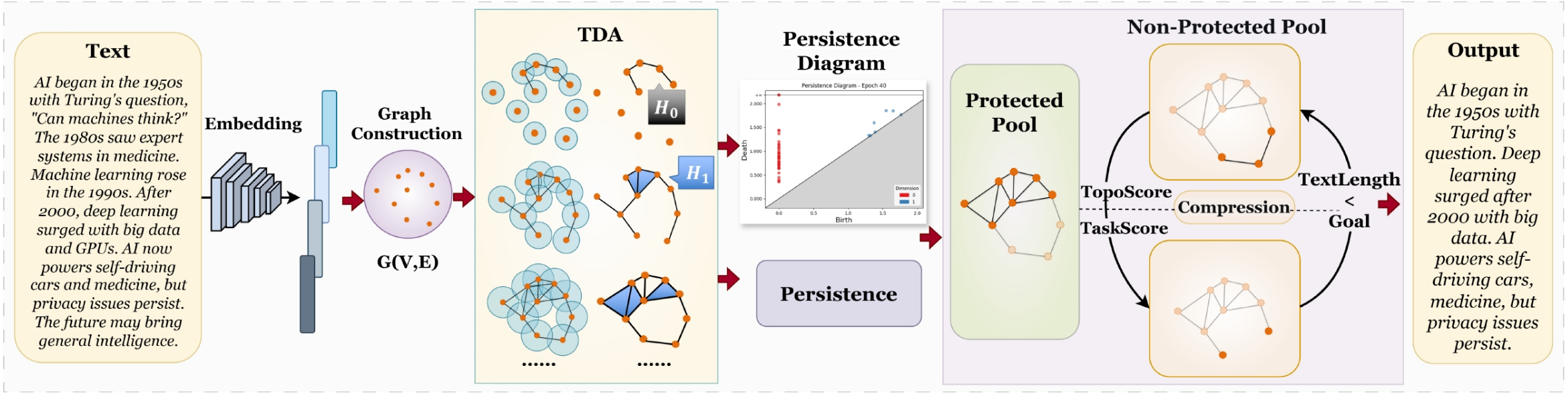

Figure 1: Overall of GloSA-sum

System Architecture

The framework consists of three main components:

- **Semantic Graph Construction**
  Sentences are encoded into embeddings and organized into a weighted graph, capturing both semantic similarity and positional relationships.

- **Topological Backbone Extraction (Protected Pool)**
  Using TDA, the model identifies:
  semantic clusters (H0) → core topics
  logical cycles (H1) → cross-paragraph dependencies
  These are preserved as the document's structural backbone.

- **Iterative Compression Module**
  Non-critical sentences are progressively removed based on structural importance and task relevance, without recomputing global structure.

2 Methodology

Key Innovation

- **Global Structure Awareness**
  Introduces TDA to explicitly model semantic and logical structures
- **Protected Pool Mechanism**
  One-time extraction of core structure avoids repeated expensive computation
- **Topology-Guided Compression**
  Uses lightweight proxy metrics instead of full recomputation
- **Hierarchical Strategy**
  Enables scalable summarization for long documents

Core Algorithm

The compression process is guided by a scoring function:

- **TopoScore**
  Measures how strongly a sentence connects to the global structure
- **TaskScore**
  Measures relevance to downstream queries
- **Final Score**
  Combines both to decide which sentences to remove first

Key Advantages

- **Structure Preservation**
  Maintains semantic integrity and logical coherence
- **Efficiency**
  Avoids repeated topological computation
- **Scalability**
  Hierarchical design supports long documents
- **Better Downstream Performance**
  Retains reasoning chains for LLM tasks

3 Experiments

Main Results

- **GloSA-sum consistently outperforms strong baselines across multiple datasets**

- **Achieves higher scores on:**
  ROUGE (R-1 / R-2 / R-L) → better content coverage & structure
  BERTScore → stronger semantic preservation
  QAFactEval → improved factual consistency

Table 1: Automatic evaluation results (ROUGE scores) across datasets.

Method	CNN/DMl			GovReport			ArXiv			PubMed		
	R-1	R-2	R-L	R-1	R-2	R-L	R-1	R-2	R-L	R-1	R-2	R-L
TextRank	33.10	12.20	29.70	53.19	23.12	49.86	33.10	8.80	30.05	38.66	15.87	34.53
Lead-3	39.94	17.46	36.06	50.94	19.53	48.45	25.53	5.98	15.22	26.38	8.73	16.60
BERTSum	41.63	19.44	40.13	-	-	-	47.10	18.20	20.80	49.10	24.30	25.70
MatchSum	44.41	20.86	40.55	-	-	-	-	-	-	41.21	14.91	36.75
MemSum	-	-	-	49.14	22.92	44.33	48.23	20.17	42.31	49.14	22.92	44.33
BART	44.16	21.28	40.09	52.24	22.09	49.99	43.84	16.55	39.86	44.61	19.37	41.01
PEGASUS	44.17	21.47	41.11	54.29	20.80	51.35	43.27	19.70	34.79	44.70	17.27	25.80
BigBird	43.83	21.11	40.74	60.64	24.81	50.01	46.63	19.02	41.77	46.32	20.65	42.33
DANCER	-	-	-	-	-	-	45.01	17.60	40.56	46.34	19.97	42.42
GloSA-sum (Ours)	44.05	21.22	41.06	55.50	26.00	51.00	47.50	20.00	42.00	49.50	22.50	44.50

Performance vs. Baselines

Outperforms:
Traditional methods (TextRank, Lead-3) → lack structure modeling
Neural models (BART, PEGASUS) → lose coherence in long texts
Long-context models (BigBird, DANCER) → weaker logical consistency

3 Experiments

Efficiency Analysis

- **Near-linear scalability for long documents**
- **Only 6–8× slower than TextRank, but much faster than generative models**
- **Avoids repeated heavy computation via one-time topology extraction**

Table 2: Theoretical efficiency comparison of summarization methods.

Method	Complexity (Time / Memory)	Parallelizability
TextRank	Graph $O(n^2)$, iteration $O(E \cdot I)$	Graph iteration parallelizable
LEAD-3	$O(n)$	Fully parallelizable
BERTSum	Encoding $O(N^2 d)$	Encoder parallelizable
MatchSum	Encoding $O(N^2)$ + candidates $K \cdot O(m^2)$	Candidate-level parallelizable
MemSum	$O(N)$ × selection steps	Encoder parallelizable, sequential in selection
BART	Encoding $O(N^2)$, decoding $O(Ld)$	Encoder parallel, decoder sequential
PEGASUS	Encoding $O(N^2)$, decoding $O(Ld)$	Encoder parallel, decoder sequential
BigBird	Sparse attention $O(N)$	Encoder parallelizable
DANCER	Segmentation $O(N \log n)$, global merge $O(k)$	Segment-level parallel, merge sequential
GloSA-sum (ours)	Graph $O(n \log n)$ (one-time), iteration $\sim O(M\, e \log n)$	Intra-/inter-segment parallelizable

Table 3: Relative runtime of different methods compared to TextRank.

Method	Relative to TextRank
TextRank	1×
LEAD-3	0.17–0.33×
BERTSum	2–3.3×
MatchSum	2.7–4×
MemSum	4–5×
BART	10–15×
PEGASUS	10–20×
BigBird	8–12×
DANCER	7–10×
GloSA-sum (ours)	6–8×

3 Experiments

Ablation Study

- Removing Protected Pool → performance drops significantly
- Removing TopoScore → weaker structure preservation
- Using only H0 (no H1) → logical relations degraded

Human Evaluation

Highest scores in:

Coherence\Informativeness\Conciseness

Table 5: Ablation experiments on the GovReport dataset and effect of hierarchical compression on short documents (CNN/DM).

Dataset	Ablation Variant	ROUGE-1	ROUGE-2	ROUGE-L
GovReport	GloSA-sum (ours)	55.5	26.0	51.0
	w/o Protected Pool	50.2	22.1	45.8
	w/o TopoScore (Random)	52.4	23.3	47.0
	w/o H1 Cycle (H0 only)	54.1	24.8	49.8
	Louvain Communities	52.9	24.1	48.3
	w/o Hierarchical	–	–	–
CNN/DM	Hierarchical (Full)	44.1	21.2	41.1
	w/o Hierarchical	44.0	21.1	40.9

Table 4: Human evaluation results on coherence, informativeness, and conciseness (1–5 scale).

Method	Coherence	Informativeness	Conciseness	Avg. Score
TextRank	3.6	2.8	3.2	3.20
LEAD-3	3.0	3.2	3.4	3.20
BERTSum	3.5	3.6	3.3	3.47
MatchSum	3.6	3.8	3.5	3.63
MemSum	3.7	3.9	3.6	3.73
BART	3.8	4.0	4.0	3.93
PEGASUS	3.9	4.1	4.1	4.03
BigBird	4.1	4.2	4.0	4.10
DANCER	4.2	4.1	4.0	4.10
GloSA-sum (ours)	**4.4**	**4.3**	**4.2**	**4.30**

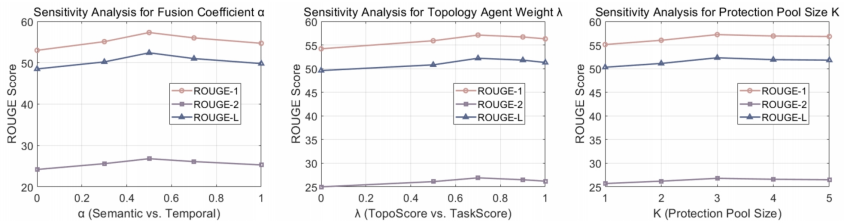

Figure 2: Hyperparameter Experiment

3 Experiments

Downstream Tasks

- Improves QA performance (e.g., SQuAD 2.0)
- Even better than using original full text in some cases

Key Takeaways

- Better structure preservation
- Strong semantic + logical integrity
- High efficiency & scalability
- Enhances LLM downstream reasoning

4 Conclusions

- Propose GloSA-sum, a global structure-aware summarization method
- Use TDA to preserve semantic and logical backbone
- Achieve better coherence, accuracy, and efficiency than existing methods
- Improve downstream reasoning performance