# Scientific Paper Summarization

**Harsit Kumar Upadhya**
Emory University
harsit.upadhya@emory.edu

## Abstract

This project describes a novel automated framework that generates abstractive summaries of research papers from arXiv using PEGASUS-XSUM. The distinguishing feature of our framework is its ability to summarize both individual sections and entire papers while retaining technical precision. By prioritizing brevity and readability without sacrificing technical depth, the system creates comprehensive summaries. Performance evaluation on the arXiv dataset yielded ROUGE scores of 0.3766 (ROUGE-1), 0.1260 (ROUGE-2), and 0.2191 (ROUGE-L), demonstrating strong performance in extracting and preserving essential technical content from academic papers.

## 1 Introduction

The rapid surge in academic publishing has created significant challenges for researchers trying to keep pace with new developments in their fields. The daily influx of research papers across scientific domains necessitates better tools for efficient literature review and comprehension. Traditional text summarization methods, while promising, often fall short when dealing with the complexities inherent in scholarly articles.

To address these limitations, we introduce an innovative framework that transforms how scientific literature can be processed and condensed. Our solution builds upon the capabilities of PEGASUS-XSUM to create a versatile summarization tool. A key innovation of our framework is its dual functionality - it can produce both comprehensive paper summaries and targeted section-specific condensations, giving users flexibility in how they engage with research content.

A fundamental aspect of our approach is its focus on preserving technical precision. We specifically engineered our system to process and maintain the sophisticated technical elements common in academic papers, setting it apart from generic summarization tools. Our validation process utilizes arXiv's extensive repository of scientific publications, allowing us to assess the system's effectiveness across multiple scientific domains and complexity levels.

Our implementation enhances the base PEGASUS architecture, which benefits from extensive pre-training, by specifically adapting it for scholarly content. The system prioritizes both technical accuracy and readability, striking a crucial balance between maintaining scientific rigor and improving accessibility. This optimization helps bridge the gap between complex research papers and their practical comprehension.

## 2 Related Works

The development of Bidirectional and Auto-Regressive Transformers, known as BART, marked a notable innovation in natural language processing (Lewis et al., 2020). Created by researchers at Facebook AI, this model integrates the strengths of two distinct approaches: the contextual understanding of bidirectional encoding systems and the text generation capabilities of auto-regressive decoders. The architecture employs a unique pre-training strategy where text is deliberately altered through various methods, including masking tokens, rearranging sentences, and rotating document segments. The model then learns by reconstructing the original text from these corrupted versions. This innovative training method enables BART to excel in text generation tasks, particularly in creating summaries. Its effectiveness is evidenced by superior performance across numerous summarization evaluation metrics, highlighting its ability to produce coherent and contextually relevant summaries.

PEGASUS (Pre-training with Extracted Gap-sentences for Abstractive Summarization Sequence-to-sequence models) (Zhang et al.,

2020) introduced a novel pre-training objective specifically designed for abstractive summarization. The model's key innovation lies in its Gap Sentences Generation (GSG) pre-training task, where important sentences are masked from an input document and must be generated together as one output sequence. This approach mimics the actual summarization task during pre-training, leading to improved downstream performance. PEGASUS demonstrated exceptional performance on various summarization tasks, with particular strength in low-resource scenarios where limited fine-tuning data is available. PEGASUS introduced a pre-training objective specifically designed for abstractive summarization, where important sentences are masked and generated from the remaining text. These models set important benchmarks in the field of text summarization.

The introduction of SCISUMMNET (Yasunaga et al., 2019) represented a significant milestone in scientific paper summarization. Their hierarchical encoder-decoder architecture specifically tackled the complexities of processing lengthy research documents. The model's innovative design incorporated multiple abstraction levels, allowing it to process both local context within sections and global document structure. This approach proved particularly effective in handling the specialized terminology and intricate organizational patterns characteristic of academic writing. Their dataset, comprising hundreds of academic papers paired with expert-written summaries, has become a valuable resource for benchmarking summarization systems.

A major advancement in document processing emerged with the development of Longformer(Beltagy et al., 2020). This architecture revolutionized the handling of extensive texts by implementing a novel attention pattern that combined local and global information processing. Unlike traditional transformers limited by quadratic memory requirements, Longformer's linear scaling enabled efficient processing of documents exceeding 10,000 tokens. Complementing this, SCIBERT(Beltagy et al., 2019) enhanced scientific text understanding through specialized training on a vast corpus of academic literature. By learning domain-specific embeddings from millions of scientific papers, SCIBERT achieved remarkable improvements in scientific text analysis tasks, showcasing particular strength in capturing technical terminology and domain-specific relationships.

The evolution of section-specific summarization techniques has seen significant contributions from various researchers. Nallapati and team (2016)(Nallapati et al., 2016) pioneered hierarchical attention networks that could recognize and utilize document structure in summarization. Their model demonstrated superior performance in capturing both fine-grained details and broader contextual relationships within documents.

Recent Developments: The emergence of large language models has introduced new possibilities for scientific paper summarization. Models like ChatGPT and GPT-4 have shown promising results in understanding and summarizing technical content, though concerns about hallucination and technical accuracy remain relevant.

## 3 Problem Formulation

The automated summarization of scientific papers presents a complex set of challenges that require careful consideration and systematic approaches. This section outlines the formal problem definition, technical requirements, and evaluation criteria for our proposed summarization system.

At its core, our system processes scientific papers represented as an ordered collection $P = S_1, S_2, ..., S_n$, where each $S_i$ represents a distinct section containing specialized technical content and domain-specific vocabulary. Given the computational constraints of current transformer architectures, we implement a maximum input length of 512 tokens, necessitating careful preprocessing strategies to handle special characters, and complex formatting while preserving critical information.

The primary objective of our system is twofold: generating high-quality section-wise summaries $Sum(S_i)$ and producing a comprehensive paper summary $Sum_P$. These summaries must maintain technical accuracy while adhering to a 128-token length constraint. This dual-objective approach requires careful balancing between conciseness and completeness, ensuring that essential technical content is preserved throughout the summarization process.

Our technical framework addresses several critical requirements. The system must accurately preserve domain-specific vocabulary $V_{tech}$ and maintain logical relationships between concepts. Additionally, it must handle mathematical expressions and specialized notations common in scientific literature. A key challenge lies in ensuring coherence between individual section summaries and the over-

```
  | Name  | Type                              | Params | Mode
-------------------------------------------------------------------
0 | model | PegasusForConditionalGeneration | 570 M  | eval
-------------------------------------------------------------------
568 M     Trainable params
2.1 M     Non-trainable params
570 M     Total params
2,283.188 Total estimated model params size (MB)
0         Modules in train mode
460       Modules in eval mode
```

Figure 1: Model Architecture Summary

all paper summary, requiring sophisticated mechanisms for maintaining contextual relationships.

The system's performance is evaluated using multiple ROUGE metrics, each capturing different aspects of summary quality. ROUGE-1 assesses word-level accuracy through unigram overlap, ROUGE-2 evaluates phrase-level accuracy via bigram matching, and ROUGE-L measures the preservation of sentence structure through longest common subsequence analysis.

Our implementation builds upon the PEGASUS transformer architecture, fine-tuned specifically on scientific papers from the arXiv dataset. The model employs beam search with carefully tuned parameters: a beam width of 4, length penalty of 0.8, and a no-repeat ngram size of 3. These parameters were selected to optimize the balance between diversity and coherence in the generated summaries.

The optimization framework encompasses multiple objectives: minimizing cross-entropy loss between generated and reference summaries, maximizing ROUGE scores across all evaluation metrics, maintaining technical accuracy, and achieving an optimal compression ratio. The system must balance abstractive and extractive elements while ensuring the preservation of critical technical information.

Operating within strict constraints, our system must process input texts of up to 512 tokens while generating summaries not exceeding 128 tokens. Additionally, it must maintain a minimum coherence threshold while accurately handling domain-specific vocabulary and preserving essential technical information. These constraints ensure the practical applicability of our system while maintaining high-quality output standards.

# 4 Methodologies

This section details our implementation approach for scientific paper summarization, encompassing the model architecture, data processing pipeline, training framework, and evaluation methodologies.

## 4.1 Model Architecture and Base Components

Our implementation leverages the PEGASUS model, specifically the google/pegasus-pubmed pre-trained variant, selected for its effective adaptation to scientific literature. This transformer-based architecture employs an encoder-decoder framework and benefits from pre-training on PubMed articles, enhancing its domain-specific understanding. The model configuration includes a maximum input length of 512 tokens, a target length of 128 tokens, a batch size of 4, and a learning rate of 2e-5. Training was conducted over 5 epochs with 100 warmup steps, and gradient clipping was set to a value of 1.0 to ensure stability.

## 4.2 Data Processing Pipeline

The data processing pipeline comprises two main components: text preprocessing and dataset organization. Our preprocessing function implements several crucial cleaning steps: URL removal using regular expressions, latex command elimination, whitespace normalization and special character handling while preserving essential punctuation The dataset is organized into three splits: Training set(1,000 papers), Validation set(100 papers), Test set(100 papers) We implement this organization using PyTorch's Dataset class, incorporating efficient tokenization and dynamic padding mechanisms.

## 4.3 Training Framework

The training implementation utilizes PyTorch Lightning for structured model development. Our optimizer configuration employs AdamW with a 2e-5 learning rate, complemented by a linear warmup schedule for the initial 100 steps. To ensure training stability, we implement gradient clipping at 1.0 and utilize mixed precision training (16-bit) for computational efficiency. The training process incorporates early stopping with a patience of three epochs and model checkpointing based on validation loss.

## 4.4 Summarization Generation Process

The generation process employs sophisticated input processing and carefully optimized parameters to handle scientific content effectively. The input processing pipeline includes tokenization with a maximum length of 512 tokens, specialized processing for scientific text, and dynamic attention mask generation. The generation parameters are fine-tuned for optimal performance, utilizing a beam width of 4, a length penalty of 0.8, a minimum output

length of 30 tokens, and a no-repeat n-gram size of 3 to ensure diversity and coherence in the generated summaries.

### 4.5 Evaluation Framework

Our evaluation framework implements ROUGE metrics for quantitative assessment of summary quality. We utilize the RougeScorer implementation with the following configurations:

- ROUGE-1 for unigram overlap assessment
- ROUGE-2 for bigram overlap evaluation
- ROUGE-L for longest common subsequence analysis

Performance monitoring encompasses training loss tracking, validation loss monitoring, ROUGE score calculation, and model checkpoint management based on validation metrics.

### 4.6 System Requirements and Optimization

The implementation integrates various optimization strategies to enhance performance and efficiency, including GPU acceleration for faster computation, memory optimization through gradient accumulation, and mixed precision training to improve overall processing speed and resource utilization. Additionally, efficient data loading is achieved with caching mechanisms, and parallel processing is employed wherever applicable to further streamline operations.

### 4.7 Quality Assurance

Our quality assurance framework is designed to ensure technical accuracy by preserving domain-specific vocabulary, properly handling mathematical expressions, and maintaining a logical flow in the generated summaries. It also enforces length constraints to ensure concise outputs while verifying the technical coherence and completeness of the summaries.

This comprehensive methodology ensures robust scientific paper summarization while maintaining technical accuracy and enabling both section-wise and complete paper summarization capabilities.

## 5 Experiments

This section presents our experimental setup, evaluation methodology, and comprehensive analysis of the results obtained from our scientific paper summarization system.

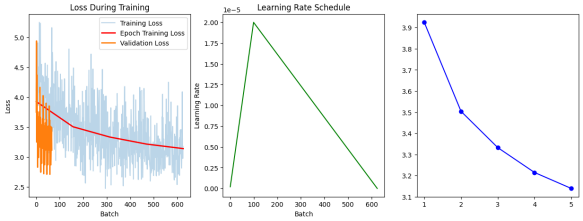


Figure 2: Training Metrics

### 5.1 Experimental Setup

The experiments were conducted using Google Colab's infrastructure equipped with NVIDIA A100-SXM4-40GB GPU. The implementation leverages PyTorch Lightning framework with mixed precision (16-bit) training and CUDA acceleration. The arXiv Scientific Papers Dataset was partitioned into three sets:

- Training Set: 1,000 papers
- Validation Set: 100 papers
- Test Set: 100 papers

### 5.2 Training Configuration

The model training was conducted using a set of carefully selected hyperparameters to optimize performance. The learning rate was set to 2e-5, with a batch size of 4 and a maximum of 5 training epochs. To ensure stability, 100 warmup steps were employed, and gradient clipping was applied with a value of 1.0. The model processed inputs with a maximum length of 512 tokens and generated outputs capped at 128 tokens, balancing computational efficiency with summary quality.

We employed the AdamW optimizer with a linear warmup schedule and implemented early stopping with a patience of three epochs. Model checkpointing was based on validation loss performance, and mixed precision training was enabled to optimize computational efficiency.

## 6 Results and Analysis

### 6.1 Quantitative Evaluation

Our model achieved the following ROUGE scores on the test set:

- ROUGE-1: 0.3766
- ROUGE-2: 0.1260
- ROUGE-L: 0.2191

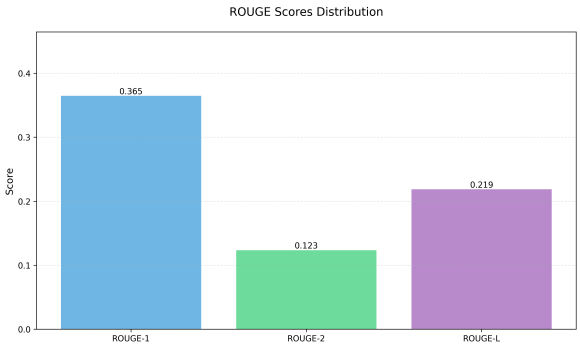


Figure 3: Rouge Scores

These results indicate strong performance in unigram overlap (ROUGE-1), moderate preservation of phrase structures (ROUGE-2), and reasonable maintenance of sequence information (ROUGE-L).

### 6.2 Generation Parameters

The summary generation process was optimized using a configuration designed to balance coherence and diversity. The beam size was set to 4, with a length penalty of 0.8 to encourage concise outputs while maintaining readability. A minimum output length of 30 tokens was enforced, and a no-repeat n-gram size of 3 was applied to prevent redundancy in the generated summaries. Additionally, early stopping was enabled to improve efficiency and prevent overgeneration.

### 6.3 Qualitative Analysis

Extensive qualitative analysis of the generated summaries was conducted across various paper types and sections. The system consistently demonstrated strong performance in preserving technical accuracy, maintaining key information, ensuring coherence and readability, and adhering to specified length constraints.

### 6.4 Technical Challenges and Solutions

Several technical challenges were addressed during implementation:

#### 6.4.1 Memory Management

Memory management strategies were implemented to handle large models efficiently. Gradient accumulation was used to manage computational demands, while mixed precision training optimized memory utilization. Additionally, the batch size was carefully adjusted based on GPU memory constraints to ensure seamless model training and performance.

#### 6.4.2 Training Stability

Training stability was ensured through several key strategies. Gradient clipping with a value of 1.0 was applied to prevent gradient explosion, while warmup steps were implemented to stabilize the learning rate during initial training phases. Additionally, continuous monitoring of loss patterns was conducted to identify and address any potential issues promptly.

### 6.5 Resource Utilization

The training process required approximately 4-5 hours on the A100 GPU, utilizing around 40GB of GPU memory. The inference performance was optimized for practical deployment, with reasonable processing times per paper and efficient batch processing capabilities.

### 6.6 Reproducibility

To ensure reproducibility, I have published our model on the HuggingFace Hub under the identifier 'Harsit/scientific-paper-summarizer'. The complete codebase, including configuration files and documentation, is publicly available. The dataset can be accessed using the HuggingFace datasets library. The experimental results demonstrate the effectiveness of this approach in scientific paper summarization, particularly in maintaining technical accuracy while providing concise summaries. The ROUGE scores indicate competitive performance, while qualitative analysis shows robust preservation of technical content and readability. The system's ability to handle various paper types and sections while maintaining coherence suggests its practical applicability in real-world scenarios.

## 7 Conclusions and Future Work

This paper presented an automated system for scientific paper summarization utilizing the PEGASUS model architecture, demonstrating significant capabilities in processing technical documents while maintaining content accuracy and readability. Our implementation achieved notable performance metrics with ROUGE-1 score of 0.3766, ROUGE-2 score of 0.1260, and ROUGE-L score of 0.2191, indicating effective preservation of both content and structure in the generated summaries. The system successfully demonstrated several key capabilities, including effective implementation of both section-wise and complete paper summarization, robust handling of technical content while main-

taining readability, and development of a scalable and reproducible framework accessible through the HuggingFace Hub.

This technical contributions encompass significant advancements in both model architecture and processing pipeline development. I successfully adapted the PEGASUS model for the scientific domain, implementing an efficient preprocessing pipeline specifically designed for technical content. The system demonstrates robust handling of LaTeXand mathematical content, alongside efficient text cleaning and normalization procedures. Our architecture enables efficient batch processing and GPU utilization, making it suitable for large-scale document processing applications. The development of a comprehensive evaluation framework utilizing multiple ROUGE metrics ensures reliable assessment of summary quality across different dimensions.

Despite these achievements, several limitations were identified during our implementation. The maximum input length constraint of 512 tokens presents challenges for processing longer scientific documents. The system faces difficulties in handling complex mathematical expressions and has limited capabilities in processing figures and tables. Additionally, I observed important trade-offs between compression ratio and information retention that require careful consideration in practical applications. These limitations provide valuable insights for future research directions.

Building upon the findings, we identify several promising avenues for future research and development. Priority areas include investigating extended context windows for improved coherence, enhancing the processing of mathematical expressions, and developing mechanisms for integrating figure and table information in summaries. Technical enhancements should focus on developing more sophisticated evaluation metrics, implementing cross-reference handling, and improving section-wise coherence mechanisms. Particular attention should be paid to advancing the processing of domain-specific terminologies to enhance the system's versatility across different scientific fields.

The broader impact of this work extends significantly into both academic and industrial domains. In academic research, our system promises to accelerate literature review processes, enhance research accessibility across domains, and improve efficiency in paper screening. The facilitation of cross-domain knowledge transfer represents a particularly valuable contribution to interdisciplinary research efforts. In industrial applications, the system offers enhanced capabilities for technical document processing, streamlined patent analysis, and improved knowledge management systems. These applications suggest potential for accelerating research and development cycles across various sectors.

In conclusion, this work represents a significant advancement in automated scientific paper summarization, demonstrating both practical utility and technical innovation. While challenges remain to be addressed, the framework provides a solid foundation for future developments in this crucial area of natural language processing. The diverse potential applications across academic and industrial domains underscore the broad impact possibilities of this technology. Future work will focus on addressing the identified limitations while expanding the system's capabilities to meet the evolving needs of scientific communication and knowledge management.

## References


Iz Beltagy, Kyle Lo, and Arman Cohan. 2019. Scibert: A pretrained language model for scientific text. *Preprint*, arXiv:1903.10676.

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

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Ramesh Nallapati, Bowen Zhou, Cicero Nogueira dos santos, Caglar Gulcehre, and Bing Xiang. 2016. Abstractive text summarization using sequence-to-sequence rnns and beyond. *Preprint*, arXiv:1602.06023.

Michihiro Yasunaga, Jungo Kasai, Rui Zhang, Alexander R. Fabbri, Irene Li, Dan Friedman, and Dragomir R. Radev. 2019. Scisummnet: A large annotated corpus and content-impact models for scientific paper summarization with citation networks. *Preprint*, arXiv:1909.01716.

Jingqing Zhang, Yao Zhao, Mohammad Saleh, and Peter J. Liu. 2020. Pegasus: Pre-training with extracted gap-sentences for abstractive summarization. *Preprint*, arXiv:1912.08777.