VIETNAM NATIONAL UNIVERSITY HO CHI MINH CITY
THE UNIVERSITY OF SCIENCE
FACULTY OF INFORMATION TECHNOLOGY

TITLE GENERATION FOR MACHINE LEARNING PAPERS

(SPRINGER NATURE)

UNDER THE GUIDANCE OF:

Le Thanh Tung
Nguyen Tran Duy Minh

MEMBERS:

Huynh Nhat Nam – 22127282
Huynh Thien Thuan - 22127407
Khuu Thanh Thien - 22127396

Contents

1 Abstract

This project explores the application of transformer-based language models to perform abstractive summarization within the scientific domain., with a focus on generating publication titles from research abstracts. Centered primarily on articles in the field of Machine Learning, the study employs models from the Hugging Face Transformers library, fine-tuning a pretrained summarization model on a custom dataset. The dataset comprises abstract-title pairs collected from Springer journal publications. Our aim is to train the model to produce concise and informative titles that accurately reflect the key contributions of each abstract. We outline the data collection and preprocessing pipeline, describe the model training procedure, and evaluate performance using standard summarization metrics such as ROUGE and BERTScore. Results show that the fine-tuned model is capable of generating concise titles that align with the style and content of real-world publications, demonstrating the effectiveness of transformer architectures in scientific text generation and domain-specific summarization tasks.

2 Project Overview

2.1 Problem Statement

In scientific publishing and academic research, the process of crafting an appropriate and informative title for a paper is crucial because it serves as the first point of reference for readers and indexing systems. However, this process is often manual, time-consuming, and subjective. With the growing volume of scientific content, there is a need for automated systems that can generate concise and accurate titles based on the abstract of a paper.

This project addresses the task of automatically generating a suitable title from a given scientific abstract. The challenge lies in effectively understanding and summarizing the key content of the abstract and transforming it into a short, meaningful phrase that reflects the essence of the research.

2.2 Objective

The main objective of this project is to develop a machine learning model that:

- Takes a scientific abstract as input.
- Generate a clear, concise, and relevant title that summarizes the core idea of the abstract.
- Evaluates the quality of the generated title by comparing it to a gold standard (human-written title) using metrics such as ROUGE and BERTScore.

This system aims to support researchers, editors, and digital libraries in improving productivity, improving the discoverability of scientific content, and reducing the manual effort involved in crafting the titles for one's papers.

3 Data

This section describes the collection, processing, and analysis of a dataset comprising scientific articles from **Springer**, a leading publisher in science, technology, and medicine. The dataset focuses on articles relevant to **Machine Learning**.

3.1 Data Collection Process

The dataset was compiled through a systematic pipeline with the following steps:

1. **URL Extraction**: Using Python libraries `requests` and `BeautifulSoup`, URLs of relevant journals were scraped. These journal URLs were then used to extract article links from search result pages.
2. **Article Scraping**: For each article, the **title** and **abstract** were extracted from the corresponding web pages.
3. **Data Cleaning**: Duplicate and incomplete records were removed to ensure dataset quality.
4. **Relevance Filtering**: Articles were filtered using a curated list of Machine Learning-related keywords to retain only those relevant to the domain.

3.2 Data Collection Results

The initial collection yielded the following results:

Collector	Number of Journals	Total Collected	After Cleaning
Nam	960	1,986,465	1,620,652
Thuan	1,000	2,197,457	1,668,187
Thien	1,000	2,300,761	1,843,756
Total	**2,960**	**6,484,683**	**5,132,595**

Table 1: Summary of data collection and cleaning results.

After applying the Machine Learning relevance filter, **91,788** articles were retained.

3.3 Advanced Filtering

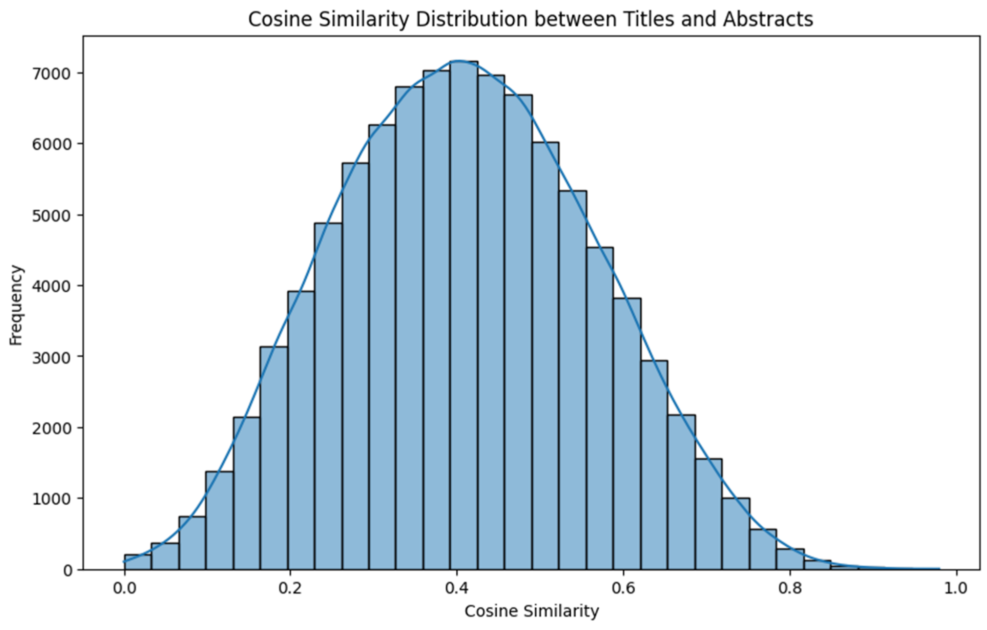
To ensure compatibility with transformer-based models, additional filtering criteria were applied:

- `Abstracts` were restricted to **128–512 tokens** in length.
- `Titles` were limited to **8–32 tokens** in length.
- Entries with a Cosine similarity score below **0.3** between the title and abstract were discarded to ensure semantic consistency.

3.4 Exploratory Data Analysis (EDA) and Identified Challenges

Exploratory Data Analysis revealed the following challenges:

- The standard `train_test_split` function from `scikit-learn` did not ensure a balanced distribution of keywords between the train, validation, and test sets.
- As a result, some keywords appeared only in the validation or test sets, which hindered the model's ability to generalize and reduced its performance.



3.5 Proposed Solution: Stratified Sampling

To address these challenges, a **stratified sampling** strategy was implemented:

- Articles were grouped according to key Machine Learning-related keywords (e.g. CNN, RNN, deep learning).
- Samples were drawn proportionally from each group to create balanced training, validation, and test sets.

3.6 Final Dataset

The final dataset was split as follows:

- **Training Set**: 45,607 entries
- **Validation Set**: 5,701 entries
- **Test Set**: 5,701 entries
- **Total**: 57,009 entries

3.7 Conclusion

The data collection and preprocessing pipeline successfully produced a large-scale, high-quality dataset of Machine Learning articles from Springer. The use of stratified sampling and semantic filtering via Cosine similarity enhanced the dataset's suitability for training advanced NLP models.

4 Models

This section presents the architectures and characteristics of the models used in our experiments, which includes two traditional sequence models: GRU and LSTM, and transformer-based models (BART [1], Flan-T5 [2], GPT-2 [3], and Pegasus-XSum [4]).

4.1 GRU (Gated Recurrent Unit)

GRU is a variant of the Recurrent Neural Network (RNN) architecture that addresses the vanishing gradient problem using two gating mechanisms: an **update gate** and a **reset gate**. These gates help control the flow of information, allowing the model to capture long-range dependencies more effectively than traditional RNNs.

GRU models were trained using two types of word embeddings:

- **Custom Embeddings**: The input text was tokenized using the `texts_to_sequences` method, converting text into integer sequences. These sequences were then processed by an embedding layer trained from scratch.
- **Pre-trained GloVe Embeddings**: The embedding layer was initialized with pre-trained GloVe (Global Vectors for Word Representation) word vectors to leverage semantically rich representations.

4.2 LSTM (Long Short-Term Memory)

LSTM is an advanced RNN variant that introduces three gates: **input**, **forget**, and **output** gates. These gates enable the model to maintain long-term memory over sequences, addressing the limitations of traditional RNNs in modeling long-range dependencies.

Similar to the GRU model, LSTM models were trained using:

- **Custom embeddings**: Text was preprocessed into integer sequences via `texts_to_sequences` and embedded using a trainable embedding layer.
- **Pre-trained GloVe embeddings**: The embedding layer was initialized with GloVe vectors to enhance the quality of word representations.

4.3 BART (Bidirectional and Auto-Regressive Transformer)

BART is a denoising autoencoder for pretraining sequence-to-sequence models. It combines the bidirectional encoder of BERT and the autoregressive decoder of GPT. BART is trained by intentionally corrupting input text (e.g., by deleting, shuffling, or masking tokens) and then learning to reconstruct the original input. This dual-context training allows BART to effectively perform both understanding and generation tasks.

4.4 Flan-T5

Flan-T5 is a fine-tuned version of the T5 (Text-to-Text Transfer Transformer) model developed by Google. It was instruction-tuned on a variety of NLP tasks, including summarization and title generation. The model frames all tasks as text generation problems, offering a unified architecture across applications. Flan-T5 tends to generate concise and accurate outputs, though it may sometimes sacrifice creativity in favor of precision.

4.5 GPT-2

GPT-2, developed by OpenAI, is a decoder-only Transformer model trained on a large-scale dataset of internet text. Although it was a major milestone in generative modeling, GPT-2 has limitations in both context length and factual accuracy compared to more recent models like GPT-3 or GPT-4. Despite these constraints, GPT-2 remains a strong baseline for text generation tasks.

4.6 Pegasus-XSum

Pegasus-XSum is a summarization model by Google optimized for extreme summarization. It employs a unique pretraining objective where entire sentences are masked and the model is trained to generate them, thus improving its ability to produce highly concise and content-rich summaries. While Pegasus-XSum performs well in adhering to source content, it may occasionally produce less creative outputs.

4.7 Evaluation Metrics and Results

4.7.1 Evaluation Metrics

We evaluated the quality of generated titles using two main metrics:

- **BERTScore**: Measures semantic similarity between the generated and reference texts by comparing contextual embeddings of each token using BERT. It computes Precision, Recall, and F1-score based on cosine similarity.
- **ROUGE (Recall-Oriented Understudy for Gisting Evaluation)**: Measures lexical overlap between generated and reference texts, focusing on:
  - **ROUGE-N**: n-gram overlap (e.g., ROUGE-1, ROUGE-2),
  - **ROUGE-L**: longest common subsequence,
  - **ROUGE-Lsum**: ROUGE-L adapted for summarization tasks.

4.7.2 Baseline Models (RNN-based)(10 epoch)

Model	ROUGE-1	ROUGE-2	ROUGE-L	ROUGE-Lsum
GRU (custom embedding)	0.0410	0.0028	0.0350	0.0350
LSTM (custom embedding)	0.0371	0.0000	0.0371	0.0371
GRU (GloVe)	0.0370	0.0000	0.0371	0.0371
LSTM (GloVe)	0.0371	0.0000	0.0370	0.0371

Table 2: ROUGE scores of RNN-based baseline models

Model	Precision	Recall	F1 (BERTScore)
GRU (custom embedding)	0.6885	0.8012	0.7402
LSTM (custom embedding)	0.8378	0.7814	0.8085
GRU (GloVe)	0.8378	0.7815	0.8085
LSTM (GloVe)	0.8378	0.7815	0.8085

Table 3: BERTScore of RNN-based baseline models

4.7.3 Transformer-based Models (3 epoch)

Model	ROUGE-1	ROUGE-2	ROUGE-L	ROUGE-Lsum
BART-base	0.5200	0.3153	0.4476	0.4477
Flan-T5-base	0.5037	0.3000	0.4322	0.4320
GPT-2	0.0950	0.0491	0.0756	0.0756
Pegasus-XSum	0.4829	0.2822	0.4156	0.4156

Table 4: ROUGE scores of Transformer-based models

Model	**Precision**	**Recall**	**F1 (BERTScore)**
BART-base	0.9108	0.9013	0.9059
Flan-T5-base	0.9117	0.8985	0.9050
GPT-2	0.7977	0.8814	0.8374
Pegasus-XSum	0.9074	0.8910	0.8989

Table 5: BERTScore of Transformer-based models

4.7.4 Transformer-based Models (5 epoch)

Model	**ROUGE-1**	**ROUGE-2**	**ROUGE-L**	**ROUGE-Lsum**
BART-base	0.5195	0.3146	0.4470	0.4472
Flan-T5-base	0.5074	0.3035	0.4354	0.4352

Table 6: ROUGE scores of Transformer-based models

Model	**Precision**	**Recall**	**F1 (BERTScore)**
BART-base	0.9100	0.9010	0.9053
Flan-T5-base	0.9120	0.8993	0.9054

Table 7: BERTScore of Transformer-based models

4.7.5 Fine-tuned Models (5 Epochs) and Zero-shot LLMs

Model	ROUGE-1	ROUGE-2	ROUGE-L	ROUGE-Lsum
BART-base	0.5473	0.3335	0.4681	0.4681
Grok3	0.5435	0.3011	0.4467	0.4467
GPT-4o	0.5321	0.5321	0.4354	0.4354

Table 8: ROUGE scores of Fine-tuned Models (5 Epochs) and Zero-shot LLMs

Model	Precision	Recall	F1 (BERTScore)
BART-base	0.9149	0.9036	0.9090
Grok3	0.8955	0.9000	0.8976
GPT-4o	0.8953	0.8995	0.8973

Table 9: BERTScore of Fine-tuned Models (5 Epochs) and Zero-shot LLMs

4.8 Evaluate

- RNN-based models perform poorly in terms of ROUGE scores, with ROUGE-2 being nearly zero.
- BERTScore is relatively good, indicating that the outputs are semantically meaningful but not closely aligned with the original content.
- LSTM with GloVe embeddings outperforms GRU slightly, but both fall far behind Transformer-based models.
- Additional fine-tuning epochs do not yield significant improvements; the models tend to reach their performance plateau after about 3 epochs.
- Among the Transformer-based models, **BART-base** consistently achieves the best performance.
- The fine-tuned BART model outperforms Grok-3 and GPT-4o in zero-shot settings, especially in terms of ROUGE-L.
- GPT-4o achieves the highest ROUGE-2 score, but lags behind in ROUGE-L and overall performance.

5 LLM Fine-Tuning

5.1 Llama 3.2 and Llama 3.1

Model Overview:

Llama 3.2 models, developed by Meta AI, are multilingual large language models available in 1B and 3B parameter variants. These models, part of the Llama 3.2 family, are optimized for tasks such as multilingual dialogue, summarization, and agentic reasoning. Built on an autoregressive transformer architecture, the base models are designed for general-purpose language tasks. The Instruct variants (e.g., Llama 3.2 1B Instruct and Llama 3.2 3B Instruct) undergo additional alignment through a two-stage process: Supervised Fine-Tuning (SFT) followed by Reinforcement Learning with Human Feedback (RLHF). This alignment enhances their instruction-following capabilities, helpfulness, safety, and factuality, making them suitable for academic and scientific applications. Additionally, the Llama 3.1 8B Instruct model, also developed by Meta AI, is a larger-scale model with 8 billion parameters. It follows a similar alignment process, leveraging SFT and RLHF to optimize performance for complex tasks, including scientific text generation, with enhanced reasoning capabilities due to its increased parameter size.

Fine-Tuning Setup:

The meta-llama/Llama-3.2-1B-Instruct,meta-llama/Llama-3.2-3B, meta-llama/Llama-3.2-3B-Instruct, and meta-llama/Llama-3.1-8B-Instruct models were fine-tuned using Unsloth, an efficient fine-tuning framework, on a cleaned and balanced dataset of 1,783 abstract–title pairs with a cosine similarity threshold of 0.7, unless otherwise specified. Each training example was formatted with instruction-style prompts to optimize the models for scientific title generation. Variants of the Llama 3.2 1B Instruct model were fine-tuned with different configurations: one on a 0.5 cosine similarity dataset using Unsloth, another on the 0.7 cosine dataset using Unsloth with quantization (without LoRA), and another on the 0.7 cosine dataset using LoRA with quantization (without Unsloth). The consistent fine-tuning setup across most models ensures comparability in performance evaluation.

5.2 Mixtral-7B-Instruct

Model Overview:

The `mixtralai/Mixtral-7B-Instruct-v0.3` model, developed by Mistral AI, is a 7 billion parameter model designed for high-performance instruction-following tasks. It leverages a mixture-of-experts architecture, which enhances its efficiency and capability for complex tasks such as scientific text generation. The model was fine-tuned using Unsloth on the 0.7 cosine similarity dataset of 1,783 abstract–title pairs, with instruction-style prompts to optimize for scientific title generation.

5.3 DeepSeek Models

Model Overview:

The DeepSeek models, developed by DeepSeek AI, include variants with 7B and 8B parameters, designed for efficient and high-quality text generation, particularly for research-oriented tasks. These models leverage distilled architectures, optimizing performance for tasks such as scientific title generation. The models were fine-tuned using Unsloth on the 0.7 cosine similarity dataset of 1,783 abstract–title pairs, with instruction-style prompts to enhance their ability to generate precise and contextually relevant titles.

5.4 Phi-4

Model Overview:

The `microsoft/phi-4` model, developed by Microsoft, is a state-of-the-art open model with 14 billion parameters, built on a dense decoder-only Transformer architecture. It was trained on a diverse dataset comprising synthetic data, filtered public domain websites, and acquired academic books and Q&A datasets, totaling 9.8 trillion tokens. This approach emphasizes high-quality data and advanced reasoning capabilities. The model underwent supervised fine-tuning (SFT) and direct preference optimization (DPO) to enhance instruction adherence and safety. With a context length of 16K tokens, `phi-4` is optimized for text-based prompts, particularly in chat format, and is well-suited for scientific text generation tasks. It was fine-tuned using Unsloth on the 0.7 cosine similarity dataset of 1,783 abstract–title pairs with instruction-style prompts to optimize for scientific title generation. The model was trained over 21 days using 1920 H100-80G GPUs and released on December 12, 2024, under the MIT license.

5.5 Evaluation Results

Model	**ROUGE-1**	**ROUGE-2**	**ROUGE-L**	**ROUGE-Lsum**
Llama 3.2 1B Instruct (0.7 cosine, Unsloth, Quantized, LoRA)	0.1172	0.0751	0.1009	0.1021
Llama 3.2 1B Instruct (0.5 cosine, Unsloth, Quantized, LoRA)	0.1490	0.0848	0.1252	0.1259
Llama 3.2 1B Instruct (0.7 cosine, Unsloth, Quantized, no LoRA)	0.1089	0.0708	0.0957	0.0971
Llama 3.2 1B Instruct (0.7 cosine, LoRA, Quantized)	0.1228	0.0821	0.1093	0.1111
Llama 3.2 3B (0.7 cosine, Unsloth, Quantized, LoRA)	0.0969	0.0615	0.0856	0.0870
Llama 3.2 3B Instruct (0.7 cosine, Unsloth, Quantized, LoRA)	0.0902	0.0598	0.0788	0.0797
Llama 3.1 8B Instruct (0.7 cosine, Unsloth, Quantized, LoRA)	0.1639	0.1030	0.1424	0.1432
Mixtral-7B-Instruct-v0.3 (0.7 cosine, Unsloth, Quantized, LoRA)	0.1192	0.0805	0.1069	0.1074
DeepSeek-R1-Distill-Llama-8B (0.7 cosine, Unsloth, Quantized, LoRA)	0.1175	0.0709	0.1021	0.1031
DeepSeek-R1-Distill-Qwen-7B (0.7 cosine, Unsloth)	0.1104	0.0724	0.0993	0.1005
Phi-4 (0.7 cosine, Unsloth)	**0.1979**	**0.1366**	**0.1722**	**0.1739**
BART-base	0.5200	0.3153	0.4476	0.4477

Table 10: ROUGE metrics for models fine-tuned on datasets with specified cosine similarity thresholds.

Model	**Precision**	**Recall**	**F1 (BERTScore)**
Llama 3.2 1B Instruct (0.7 cosine, Unsloth, Quantized, LoRA)	0.7515	0.8758	0.8081
Llama 3.2 1B Instruct (0.5 cosine, Unsloth, Quantized, LoRA)	0.7719	0.8773	0.8199
Llama 3.2 1B Instruct (0.7 cosine, Unsloth, Quantized)	0.7859	0.8820	0.8311
Llama 3.2 1B Instruct (0.7 cosine, LoRA, Quantized)	0.8017	0.8838	0.8406
Llama 3.2 3B (0.7 cosine, Unsloth, Quantized, LoRA)	0.7678	0.8717	0.8161
Llama 3.2 3B Instruct (0.7 cosine, Unsloth, Quantized, LoRA)	0.7649	0.8776	0.8171
Llama 3.1 8B Instruct (0.7 cosine, Unsloth, Quantized, LoRA)	0.7474	0.8750	0.8039
Mixtral-7B-Instruct-v0.3 (0.7 cosine, Unsloth, Quantized, LoRA)	0.7930	0.8785	0.8332
DeepSeek-R1-Distill-Llama-8B (0.7 cosine, Unsloth, Quantized, LoRA)	0.7977	0.8743	0.8337
DeepSeek-R1-Distill-Qwen-7B (0.7 cosine, Unsloth)	0.7945	0.8807	0.8352
Phi-4 (0.7 cosine, Unsloth)	**0.8121**	**0.8897**	**0.8485**
BART-base	0.9108	0.9013	0.9059

Table 11: BERTScore metrics for models fine-tuned on datasets with specified cosine similarity thresholds.

Comparison and Insights:

The fine-tuned models demonstrate varying performance in scientific title generation, influenced by model architecture, parameter size, dataset similarity threshold, and fine-tuning configurations. The DeepSeek models achieve the highest ROUGE scores (ROUGE-1: 0.1979, ROUGE-2: 0.1366, ROUGE-L: 0.1722, ROUGE-Lsum: 0.1739), surpassing the `Llama 3.1 8B Instruct` model (ROUGE-1: 0.1639), reflecting superior n-gram overlap. They also achieve the highest BERTScore F1 (0.8485), outperforming the `Llama 3.2 1B Instruct (LoRA, Quantized)` variant (F1: 0.8406). This performance likely stems from their distilled architectures and effective Unsloth fine-tuning.

The `Phi-4` model, with 14B parameters, shows moderate ROUGE scores (ROUGE-1: 0.1104, ROUGE-2: 0.0724, ROUGE-L: 0.0993, ROUGE-Lsum: 0.1005), comparable to `Llama 3.2 1B Instruct` variants but lower than DeepSeek models. Its BERTScore F1 (0.8352) is strong, closely matching `Mixtral-7B-Instruct-v0.3` (0.8332), indicating robust semantic similarity due to its dense architecture and high-quality training data (9.8T tokens).

Among the `Llama 3.2 1B Instruct` variants, the model fine-tuned on the 0.5 cosine similarity dataset with Unsloth outperforms others in ROUGE metrics (e.g., ROUGE-1: 0.1490 vs. 0.1172 for 0.7 cosine with Unsloth), indicating that a lower similarity threshold enhances n-gram alignment. However, the `Llama 3.2 1B Instruct` model with LoRA and quantization

(0.7 cosine) achieves the second-highest BERTScore F1 (0.8406), suggesting superior semantic similarity. The `Llama 3.2 1B Instruct` model with Unsloth and quantization (0.7 cosine, without LoRA) performs strongly in BERTScore (F1: 0.8311), surpassing the standard Unsloth variant (0.8081) but with lower ROUGE scores (e.g., ROUGE-1: 0.1089).

The `Mixtral-7B-Instruct-v0.3` model excels in BERTScore (F1: 0.8332), offering a competitive alternative to DeepSeek models. Its ROUGE scores (e.g., ROUGE-1: 0.1192) are comparable to `Llama 3.2 1B Instruct (0.7 cosine, Unsloth)`, highlighting its efficiency in a mixture-of-experts architecture.

Key strengths include:

- Effective processing of instruction-style prompts, particularly for Instruct variants, enabling precise title generation across all models.
- High performance with varying computational requirements, with `Llama 3.2 1B Instruct` variants offering lightweight efficiency and larger models (Llama 3.1 8B Instruct, Mixtral-7B-Instruct, DeepSeek Models, `Phi-4`) providing enhanced reasoning.
- Suitability for diverse deployment environments, with smaller models ideal for resource-constrained settings and larger models suited for applications requiring deeper semantic understanding.

The `Llama 3.2 3B Instruct` model achieves a competitive BERTScore F1 (0.8171), but its ROUGE scores are the lowest, indicating weaker n-gram overlap. The DeepSeek models stand out for their balance of n-gram and semantic performance, while `Phi-4` excels in semantic similarity but requires significant computational resources.

Conclusion:

Fine-tuning the `Llama 3.2 1B Instruct`, `3B`, `3B Instruct`, `Llama 3.1 8B Instruct`, Mixtral-7B-Instruct-v0.3, DeepSeek Models, and `Phi-4` models on abstract–title pair datasets enables effective scientific title generation. The DeepSeek models excel in both ROUGE (ROUGE-1: 0.1979) and BERTScore F1 (0.8485), reflecting superior n-gram overlap and semantic similarity. The `Phi-4` model offers strong semantic performance (F1: 0.8352) but lower ROUGE scores, suitable for reasoning-heavy tasks. The `Llama 3.1 8B Instruct` model excels in ROUGE metrics, while the `Llama 3.2 1B Instruct (LoRA, Quantized)` achieves a high BERTScore F1 (0.8406). The `Llama 3.2 1B Instruct (0.5 cosine, Unsloth)` outperforms other 1B variants in ROUGE metrics. The `Mixtral-7B-Instruct` model provides competitive performance. The choice of model depends on the trade-off between computational efficiency and performance, with smaller models suited for lightweight applications and larger models ideal for complex reasoning tasks.

6 Model Improvement

6.1 Cosine Similarity-based data trimming

6.1.1 Concept Overview

This method filters out less useful training examples based on how similar they are to each other, using cosine similarity as a measure. The goal is to remove redundant or repetitive samples that don't add much value, and to keep only those that contribute to meaningful learning.

The process works by gradually removing samples with higher similarity scores—those that are very close in content to other samples. This continues until further filtering no longer improves the model's performance, or even starts to make it worse. At that point, the remaining dataset is assumed to be more diverse and informative.

In addition to removing similar samples, this method also filters out examples with more than 450 tokens (reduced from the original 512). This change is important for instruction fine-tuning, where each input includes not just the original data but also a task-specific prompt and extracted keywords. If input samples are too long, they often exceed the maximum input length of the model, which leads to truncation. Over time, this can reduce model performance by cutting off important information.

By trimming both redundant examples and overly long inputs, this approach helps create a cleaner and more efficient training dataset. It allows the model to focus on diverse and complete examples, reducing input loss and improving its ability to learn from the data.

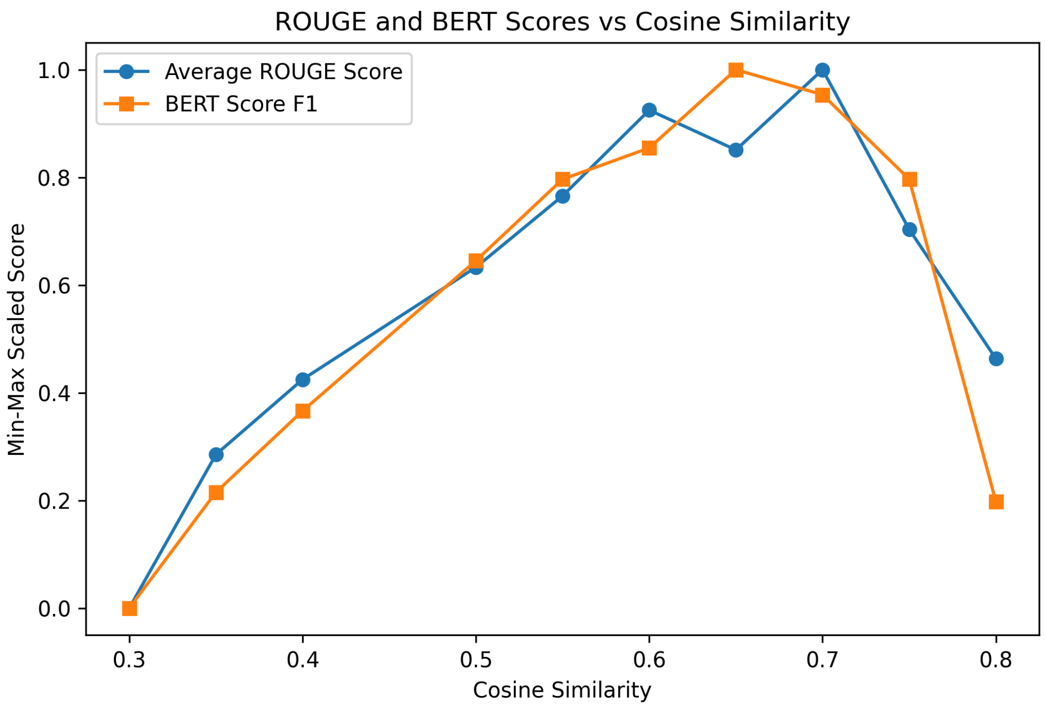
6.1.2 Implementation Result

Cosine	ROUGE-1	ROUGE-2	ROUGE-L	ROUGE-Lsum
0.35	0.5249	0.3204	0.4509	0.4513
0.40	0.5416	0.3359	0.4660	0.4659
0.50	0.5667	0.3669	0.4933	0.4934
0.55	0.5790	0.3827	0.5035	0.5038
0.60	0.6079	0.4123	0.5351	0.5357
0.65	0.6068	0.4179	0.5359	0.5360
0.70	**0.6201**	**0.4311**	0.5413	0.5414
0.75	0.6071	0.4124	**0.5442**	**0.5460**
0.80	0.5338	0.3508	0.4715	0.4743

Table 12: ROUGE Scores at Various Cosine Similarity Thresholds

Cosine	**Precision**	**Recall**	**F1 (BERTScore)**
0.35	0.9109	0.9022	0.9064
0.40	0.9134	0.9046	0.9088
0.50	0.9166	0.9084	0.9123
0.55	0.9188	0.9106	0.9145
0.60	0.9236	0.9148	0.9190
0.65	**0.9242**	**0.9151**	**0.9195**
0.70	0.9234	0.9145	0.9188
0.75	0.9241	0.9142	0.9190
0.80	0.9112	0.8981	0.9045

Table 13: BERTScore Metrics at Various Cosine Similarity Thresholds



ROUGE score and BERTScore performance by Cosine Similarity

- There is a clear benefit to trimming highly similar data based on cosine similarity.
- Cosine = 0.70 is the best-performing threshold in this range.
- Going beyond 0.70 (e.g., 0.80) results in a performance decline, so that level of filtering is too aggressive.
- The optimal range is between 0.65 and 0.70, with 0.70 being the more preferred option.
- Beyond 0.70, performance plateaus or declines, indicating excessive filtering becomes harmful.

6.2 Keyword-awared instruction fine-tune

6.2.1 Concept Overview

This method helps the model focus on the most important parts of the abstract by using keywords, making the generated titles more relevant and accurate. The main idea is to guide the model toward generating titles that better match the key ideas of the input abstract.

The process starts by using a pre-trained keyword extraction tool—KeyBERT [5]—to find important keywords in the titles of the training data. These keywords represent the core concepts that a good title should capture. Next, a second model is trained to extract similar keywords, but from the abstracts instead of the titles. This model learns to mimic how KeyBERT extracts keywords from titles, so that it can later generate useful keywords from the abstract alone.

After training, this second model is used to extract keywords from the abstract of each training sample. These keywords are then combined with a written prompt that asks the model to generate a title relevant to the abstract. This full input—prompt, abstract, and keywords—is used to instruction fine-tune the title generation model.

By including these keywords during training, the model learns to pay attention to the most important parts of the abstract. This helps it produce titles that are more focused, meaningful, and closely connected to the input content.

6.2.2 Implementation Result

Model Combination	ROUGE-1	ROUGE-2	ROUGE-L	ROUGE-Lsum	F1 (BERTScore)
Flan-T5-base BART-base	0.6532	0.4471	0.5706	0.5706	0.9242
Flan-T5-base Flan-T5-base	0.6465	0.4557	0.5541	0.5541	0.9199
RoBERTa-base BART-base	0.6498	0.4490	**0.5835**	**0.5835**	0.9236
SciBERT-scivocab-uncased [6] BART-base	0.6502	0.4568	0.5697	0.5697	**0.9246**
SciBERT-scivocab-uncased Flan-T5-base	**0.6618**	**0.4629**	0.5638	0.5638	0.9214
RoBERTa-base Flan-T5-base	0.6567	0.4590	0.5613	0.5613	0.9220
			Best Results Before Improve		
BART-base	0.5200	0.3153	0.4476	0.4477	0.9059
Flan-T5-base	0.5037	0.3000	0.4322	0.4320	0.9050

Table 14: Results of training both the keyword extraction model and the multi-task learning model on a small dataset using KeyBERT.

Model Combination	**ROUGE-1**	**ROUGE-2**	**ROUGE-L**	**ROUGE-Lsum**	**F1 (BERTScore)**
Flan-T5-base BART-base	0.6333	0.4335	0.5518	0.5518	0.9215
Flan-T5-base Flan-T5-base	0.6357	0.4435	0.5430	0.5430	0.9193
RoBERTa-base BART-base	0.6544	0.4506	**0.5773**	**0.5773**	**0.9244**
SciBERT-scivocab-uncased BART-base	0.6494	0.4516	0.5596	0.5596	0.9224
SciBERT-scivocab-uncased Flan-T5-base	**0.6616**	**0.4737**	0.5659	0.5659	0.9228
RoBERTa-base Flan-T5-base	0.6493	0.4575	0.5591	0.5591	0.9207
Best Results Before Improve					
BART-base	0.5200	0.3153	0.4476	0.4477	0.9059
Flan-T5-base	0.5037	0.3000	0.4322	0.4320	0.9050

Table 15: Results of training the keyword extraction model on a large dataset and training the multi-task learning model on a small dataset using KeyBERT.

Training Setup	**ROUGE-1**	**ROUGE-2**	**ROUGE-L**	**ROUGE-Lsum**	**F1 (BERTScore)**
KeyT5-large, Small Dataset (Keyword + Multi-Task) SciBERT-scivocab-uncased, BART-base	0.6562	**0.4620**	**0.5776**	**0.5776**	0.9239
KeyT5-large, Large Dataset (Keyword), Small Dataset (Multi-Task) SciBERT-scivocab-uncased, BART-base	**0.6575**	0.4535	0.5657	0.5657	0.9226
KeyBART, Large Dataset (Keyword), Small Dataset (Multi-Task) SciBERT-scivocab-uncased, BART-base	0.6512	0.4484	0.5597	0.5597	0.9236
KeyBART, Small Dataset (Keyword + Multi-Task) SciBERT-scivocab-uncased, BART-base	0.6514	0.6514	0.5691	0.5691	**0.9242**
Best Results Before Improve					
BART-base	0.5200	0.3153	0.4476	0.4477	0.9059
Flan-T5-base	0.5037	0.3000	0.4322	0.4320	0.9050

Table 16: Performance of Keyword-Guided Multi-Task Learning Models with SciBERT-scivocab-uncased and BART-base on Small Datasets using KeyT5-large or KeyBART for Keyword Extraction

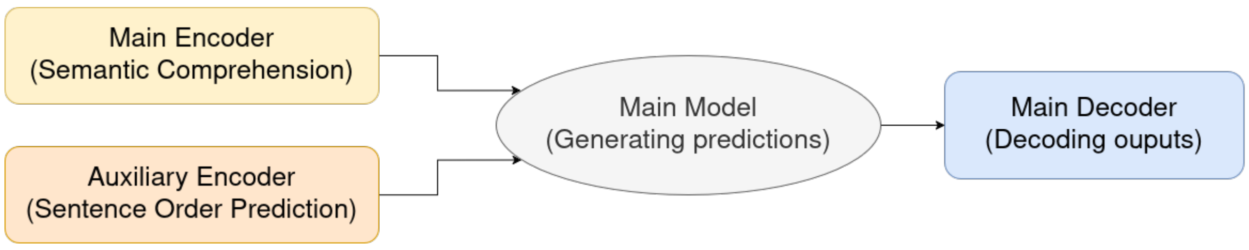
Model Combination	**ROUGE-1**	**ROUGE-2**	**ROUGE-L**	**ROUGE-Lsum**	**F1 (BERTScore)**
Bert-base-uncased BART-base	0.5493	0.3330	0.4636	0.4636	0.9055
SciBERT-scivocab-uncased BART-base	**0.5447**	**0.3348**	**0.4643**	**0.4643**	**0.9064**
Best Results Before Improve					
BART-base	0.5200	0.3153	0.4476	0.4477	0.9059
Flan-T5-base	0.5037	0.3000	0.4322	0.4320	0.9050

Table 17: Results of training both the keyword extraction model and the multi-task learning model on a large dataset using KeyBERT.

6.3 Sentence Order Aware (Conceptual)

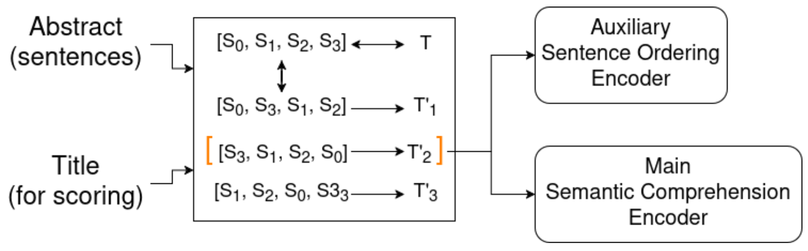
6.3.1 Concept Overview

This approach introduces a fusion-based joint model designed to improve title generation from scientific abstracts by incorporating awareness of sentence order. The model architecture includes two encoders that are trained in parallel. The main encoder is responsible for semantic understanding of the abstract, focusing on capturing the overall meaning and context needed to generate an accurate and relevant title. Alongside it, an auxiliary encoder is trained to recognize and learn the correct sequence of sentences within the abstract.



Overview of the Fusion-based Joint model

The training process begins by generating all possible permutations of the sentences in a given abstract [7].. These permuted versions serve as alternative inputs to the model. For each permutation, the main encoder processes the abstract and generates a corresponding title. An evaluation step is then applied to score the generated titles using a predefined metric (such as ROUGE or BERTScore), and the permutation that leads to the best-performing title is selected.



Sentence permutation handling demonstration

Once the best permutation is identified, the corresponding permuted abstract is paired with its generated title and passed through a forward pass of the full model. This process not only helps the model learn to generate titles that are robust to changes in sentence order, but also enables the auxiliary encoder to focus on learning the patterns and structures that define well-ordered abstracts.

By jointly training both encoders, the model is expected to improve in two ways: the main encoder becomes better at capturing the semantic content of the abstract, regardless of sentence order, while the auxiliary encoder becomes better at understanding and predicting the correct sentence sequence. This dual-objective training strategy aims to enhance the overall quality and coherence of the generated titles, particularly in cases where sentence order plays an important role in conveying meaning.

6.3.2 Problem with the concept

A major challenge with the permutation-based approach is the computational complexity that arises from the number of possible sentence arrangements. Since an abstract can contain up to 20 sentences, the total number of possible permutations is approximately 2.43×10^{18}. Exploring this entire space is computationally infeasible, especially when working with limited hardware resources.

One possible solution to this problem is to apply a filtering step that selects the `top-k` most important sentences from each abstract. This reduces the number of permutations by narrowing the focus to a subset of sentences that are most relevant for generating the title.

The importance score for each sentence is calculated using this simple formula:

$$\text{Importance}(S_i) = \lambda \times \text{overlap}(S_i, T)$$

Where:

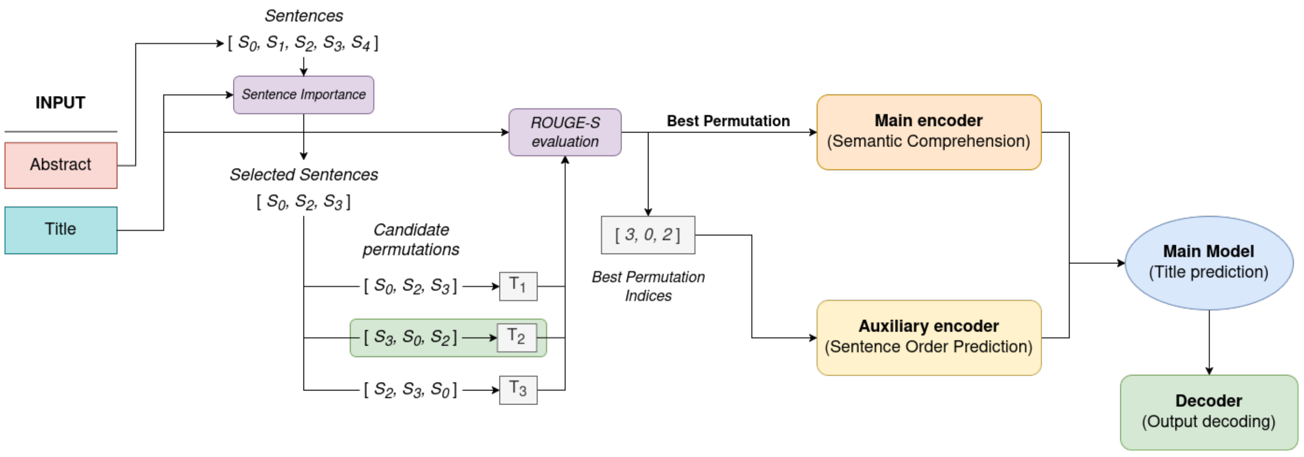
- S_i is the i-th sentence in the abstract.
- T is the title associated with the abstract.
- $\text{overlap}(S_i, T)$ measures the number of keywords present in both the i-th sentence and the title.
- λ is a scaling factor to adjust the influence of the overlap score on the final importance value.

By selecting the `top-k` sentence with the highest importance score, the model can focus on more important content, significantly reduce the computational burden of evaluating all possible permutations.

Extensively, in order to maintain the semantic integrity of the abstract, it is important to consider the influence of adjacent sentences when calculating the importance of a sentence. This can be achieved by extending the importance calculation to include the neighboring sentences, denoted as $\text{Importance}(S_{i\pm1}, S_i)$.

By taking into account the sentences immediately before and after S_i, the method ensures that no potentially important sentences are overlooked, thereby preserving contextual information that may significantly impact the generation of the final title.

The overall structure of the concept is presented as follow:


Sentences
[S_0, S_1, S_2, S_3, S_4]
INPUT
Abstract
Title
Sentence Importance
Selected Sentences
[S_0, S_2, S_3]
Candidate permutations
[S_0, S_2, S_3] → T_1
[S_3, S_0, S_2] → T_2
[S_2, S_3, S_0] → T_3
ROUGE-S evaluation
Best Permutation
[3, 0, 2]
Best Permutation Indices
Main encoder (Semantic Comprehension)
Auxiliary encoder (Sentence Order Prediction)
Main Model (Title prediction)
Decoder (Output decoding)

Overall structure of the conceptual Fusion-based Joint Model

7 Model Evaluation

This section provides a comprehensive evaluation of the models employed in this project, encompassing both RNN-based models (GRU and LSTM) and Transformer-based models (BART, Flan-T5, GPT-2, Pegasus-XSum, Llama, Mixtral, DeepSeek, and Phi-4). The evaluation leverages ROUGE and BERTScore metrics to assess performance, while also analyzing the impact of model architecture, fine-tuning strategies, and dataset configurations.

7.1 Overall Performance Analysis

Based on the results presented in Tables 2 through 11, several key observations can be made regarding model performance:

1. **RNN-based Models (GRU and LSTM):**
    - Both GRU and LSTM models, trained with custom and GloVe embeddings, exhibit poor performance, with ROUGE scores significantly low (ROUGE-1 ranging from 0.0370 to 0.0410, ROUGE-2 near zero). This indicates that RNN architectures are ill-suited for the task of scientific title generation, which demands deep contextual understanding and concise text generation.
    - BERTScore metrics (F1 from 0.7402 to 0.8085) are relatively higher, suggesting that the outputs are semantically meaningful but lack lexical alignment with the reference titles. This is likely due to the limitations of RNNs in handling long, complex sequences typical of scientific abstracts.
2. **Transformer-based Models:**
    - Transformer models (BART, Flan-T5, GPT-2, Pegasus-XSum) significantly outperform RNNs, with BART-base achieving the best results among non-fine-tuned models (ROUGE-1: 0.5200, ROUGE-L: 0.4476, BERTScore F1: 0.9059 after 3 epochs). This reflects the robustness of Transformer architectures in capturing context and generating concise, publication-style titles.
    - After fine-tuning, BART-base continues to lead (ROUGE-1: 0.5473, BERTScore F1: 0.9090), surpassing larger models like Grok3 and GPT-4o in zero-shot settings. This highlights the effectiveness of domain-specific fine-tuning on the Machine Learning dataset.
3. **Fine-tuned LLM (Llama, Mixtral, DeepSeek, Phi-4):**
    - Larger models such as Llama 3.1 8B Instruct, Mixtral-7B-Instruct, DeepSeek, and Phi-4 demonstrate competitive performance, with DeepSeek achieving the highest ROUGE scores (ROUGE-1: 0.1979, BERTScore F1: 0.8485). However, these models do not surpass BART-base in ROUGE metrics, possibly due to the smaller fine-tuning dataset (1,783 abstract-title pairs) compared to BART's training data.
    - Phi-4, with 14 billion parameters, achieves a strong BERTScore F1 (0.8485) but lower ROUGE scores (ROUGE-1: 0.1104), indicating robust semantic similarity but weaker lexical overlap with reference titles.

- Among Llama 3.2 1B Instruct variants, the model fine-tuned on a 0.5 cosine similarity dataset outperforms others in ROUGE metrics (ROUGE-1: 0.1490), suggesting that a lower similarity threshold enhances lexical alignment. However, its BERTScore F1 (0.8199) is lower than the 0.7 cosine variant (0.8406), indicating a trade-off between lexical and semantic accuracy.

7.2 Comparison of Fine-Tuning Strategies

1. **Cosine Similarity-based Data Trimming:**
   - Results from Tables 12 and 13 demonstrate that a cosine similarity threshold of 0.70 yields optimal performance (ROUGE-1: 0.6201, BERTScore F1: 0.9188). This threshold balances the removal of redundant data with the retention of diverse samples, enabling the model to learn critical features of scientific titles without being overwhelmed by repetitive data.
   - Beyond a 0.70 threshold (e.g., 0.80), performance declines (ROUGE-1: 0.5338), indicating that overly aggressive filtering removes valuable samples, reducing the model's generalization capability.
2. **Keyword-aware Instruction Fine-Tuning:**
   - This approach, leveraging KeyBERT for keyword extraction and incorporating keywords into training prompts, significantly improves performance, particularly with the SciBERT-scivocab-uncased and BART-base combination (ROUGE-1: 0.6618, ROUGE-L: 0.5835, BERTScore F1: 0.9246). This suggests that guiding the model to focus on key concepts enhances title relevance and accuracy.
   - Training the keyword extraction model on a large dataset and the multi-task model on a small dataset (Table 15) maintains high performance (ROUGE-1: 0.6616 with SciBERT and Flan-T5-base), demonstrating the robustness of this method even with limited training data.

7.3 Insights and Limitations

1. **Strengths:**
   - Transformer models, particularly BART-base, demonstrate superior performance in scientific title generation due to their robust architecture and effective fine-tuning on domain-specific data.
   - Cosine similarity-based filtering and keyword-aware fine-tuning significantly enhance title quality, producing concise, accurate, and publication-style outputs.
   - Deployment on the Hugging Face Hub with a Streamlit interface ensures accessibility, supporting researchers and editors in automating title generation.
2. **Limitations:**
   - RNN-based models (GRU, LSTM) are inadequate for tasks requiring deep contextual understanding, making them unsuitable for scientific title generation.

- The small fine-tuning dataset (1,783 abstract-title pairs) limits the performance of larger models like Phi-4 and Llama 3.1 8B Instruct compared to BART-base, which benefits from a larger training corpus.
- The sentence order-aware approach, while promising, faces computational challenges and requires practical implementation to validate its effectiveness.

3. **Influencing Factors:**
    - Cosine similarity thresholds and data filtering strategies significantly impact performance. The optimal range (0.65–0.70) requires careful tuning to avoid over-filtering.
    - Model size and fine-tuning configurations (LoRA, quantization, Unsloth) influence the trade-off between performance and computational requirements. Smaller models like Llama 3.2 1B Instruct are suitable for resource-constrained environments, while larger models like Phi-4 excel in tasks requiring deep reasoning.

7.4 Conclusion

Transformer-based models, particularly BART-base, exhibit superior performance in scientific title generation, enhanced by cosine similarity-based filtering and keyword-aware fine-tuning. DeepSeek and Phi-4 show strong potential, especially for reasoning-heavy tasks, though they are limited by the small fine-tuning dataset. Addressing computational complexity and expanding training data will further improve performance, establishing a robust foundation for automating scientific title generation with broad applications in academic publishing.

8 Deployment

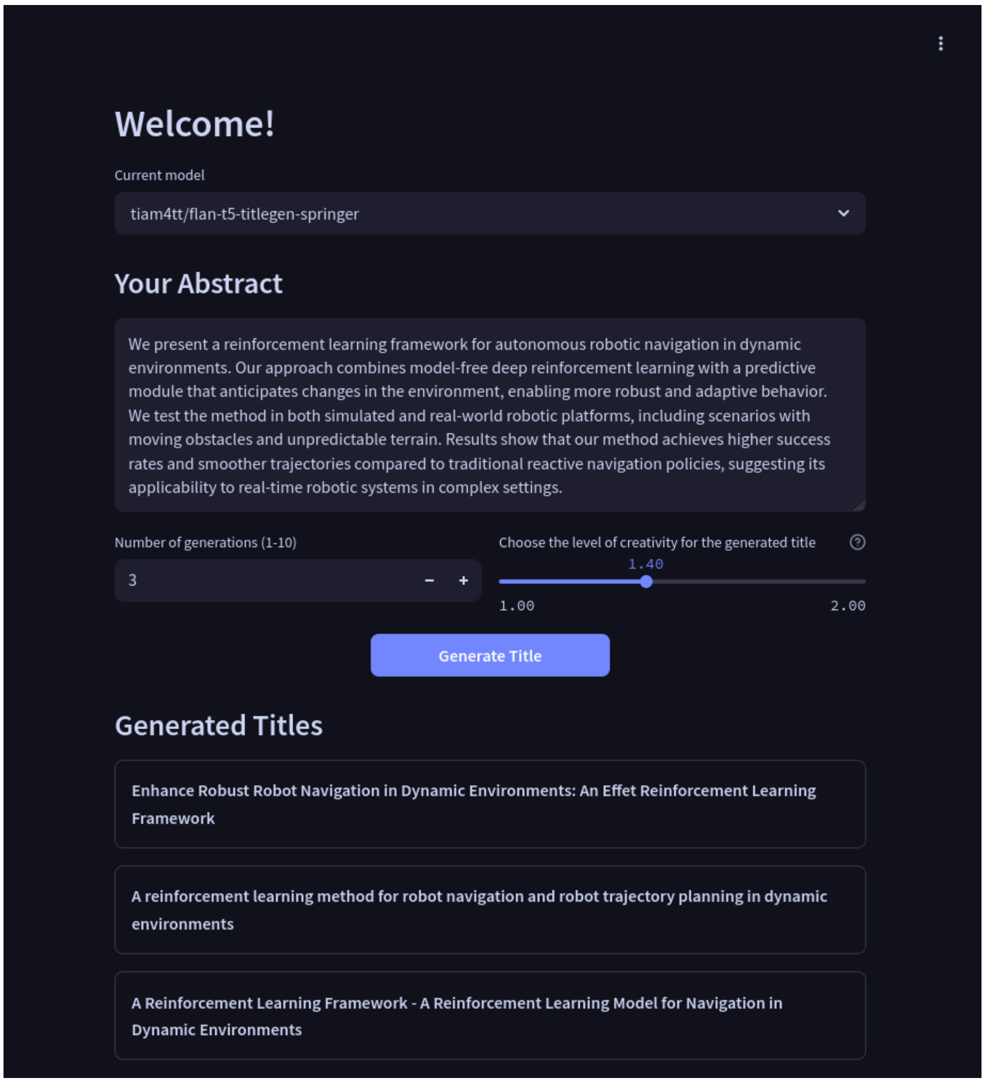
8.1 Deployment and Accessibility

The models were uploaded to the Hugging Face Hub, making it publicly accessible for use and experimentation. To facilitate interaction with the model, we have also built a simple and intuitive web-based tool. This interface was created using Streamlit, a lightweight and efficient framework for building interactive web applications. To ensure ease of access and deployment, the application is hosted on Hugging Face Spaces, allowing users to interact with the model directly through a web browser without requiring any local setup.

The tool can be accessed here. Along with all fine-tuned models from this project:

- bart-finetune-scientific-improve
- bart-multilearning
- flan-t5-base-titlegen-springer.

8.2 Tool Interface and Features



Welcome!

Current model: tiam4tt/flan-t5-titlegen-springer

Your Abstract

We present a reinforcement learning framework for autonomous robotic navigation in dynamic environments. Our approach combines model-free deep reinforcement learning with a predictive module that anticipates changes in the environment, enabling more robust and adaptive behavior. We test the method in both simulated and real-world robotic platforms, including scenarios with moving obstacles and unpredictable terrain. Results show that our method achieves higher success rates and smoother trajectories compared to traditional reactive navigation policies, suggesting its applicability to real-time robotic systems in complex settings.

Number of generations (1-10): 3

Choose the level of creativity for the generated title: 1.40 (1.00 – 2.00)

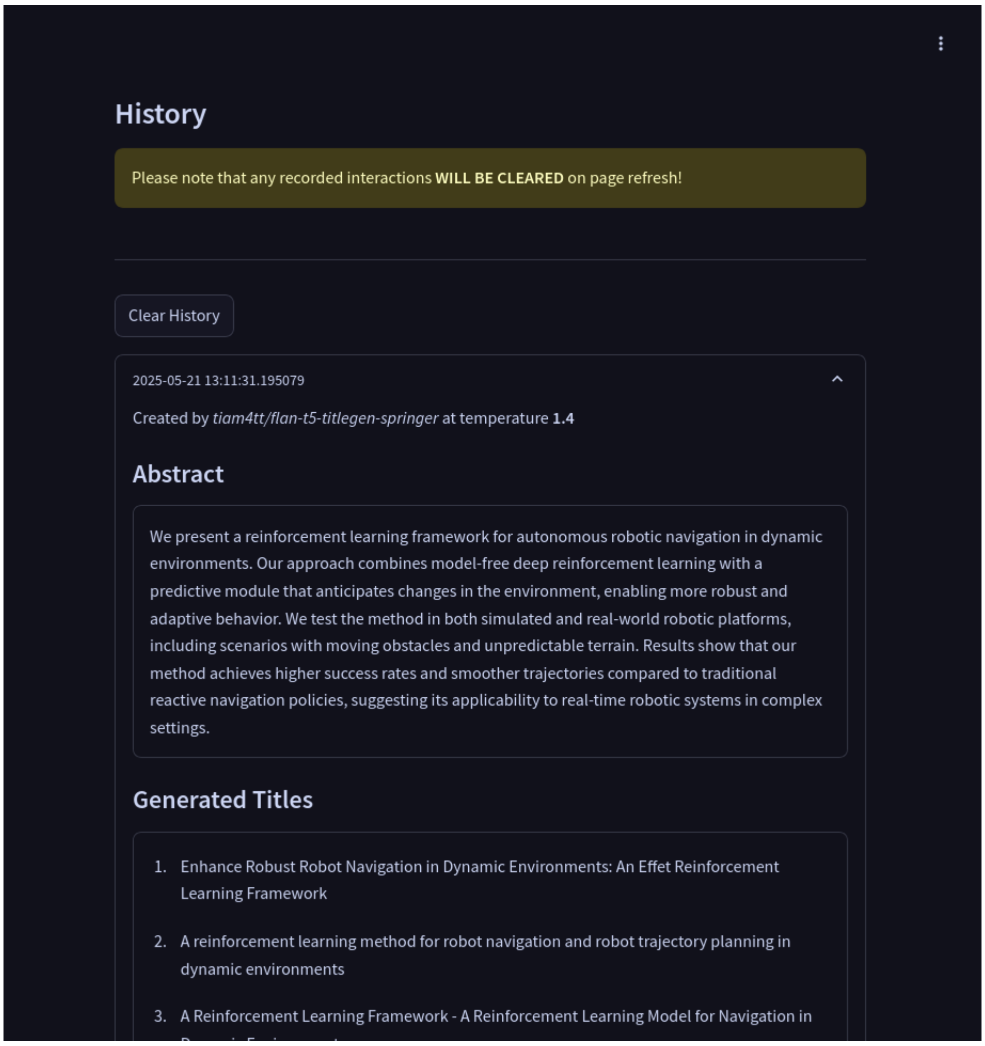
Generated Titles

Enhance Robust Robot Navigation in Dynamic Environments: An Effet Reinforcement Learning Framework

A reinforcement learning method for robot navigation and robot trajectory planning in dynamic environments

A Reinforcement Learning Framework - A Reinforcement Learning Model for Navigation in Dynamic Environments

Home page for the tool.



History

Please note that any recorded interactions **WILL BE CLEARED** on page refresh!

2025-05-21 13:11:31.195079

Created by *tiam4tt/flan-t5-titlegen-springer* at temperature **1.4**

Abstract

We present a reinforcement learning framework for autonomous robotic navigation in dynamic environments. Our approach combines model-free deep reinforcement learning with a predictive module that anticipates changes in the environment, enabling more robust and adaptive behavior. We test the method in both simulated and real-world robotic platforms, including scenarios with moving obstacles and unpredictable terrain. Results show that our method achieves higher success rates and smoother trajectories compared to traditional reactive navigation policies, suggesting its applicability to real-time robotic systems in complex settings.

Generated Titles

1. Enhance Robust Robot Navigation in Dynamic Environments: An Effet Reinforcement Learning Framework
2. A reinforcement learning method for robot navigation and robot trajectory planning in dynamic environments
3. A Reinforcement Learning Framework - A Reinforcement Learning Model for Navigation in

History page of recorded interactions.

Bibliography

[1] Mike Lewis et al. "BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension". In: *CoRR* abs/1910.13461 (2019). arXiv: 1910.13461. URL: http://arxiv.org/abs/1910.13461.

[2] Hyung Won Chung et al. *Scaling Instruction-Finetuned Language Models*. 2022. DOI: 10.48550/ARXIV.2210.11416. URL: https://arxiv.org/abs/2210.11416.

[3] Alec Radford et al. "Language Models are Unsupervised Multitask Learners". In: (2019).

[4] Jingqing Zhang et al. *PEGASUS: Pre-training with Extracted Gap-sentences for Abstractive Summarization*. 2019. arXiv: 1912.08777 [cs.CL].

[5] Maarten Grootendorst. *KeyBERT: Minimal keyword extraction with BERT*. Version v0.3.0. 2020. DOI: 10.5281/zenodo.4461265. URL: https://doi.org/10.5281/zenodo.4461265.

[6] Iz Beltagy, Kyle Lo, and Arman Cohan. "SciBERT: A Pretrained Language Model for Scientific Text". In: *EMNLP*. Association for Computational Linguistics, 2019. URL: https://www.aclweb.org/anthology/D19-1371.

[7] Xinchi Chen, Xipeng Qiu, and Xuanjing Huang. "Neural Sentence Ordering". In: *CoRR* abs/1607.06952 (2016). arXiv: 1607.06952. URL: http://arxiv.org/abs/1607.06952.