Highlight Generation from Scientific Papers Using XGBoost Regressor Model

Papun Roy[1,*,†], Kamal Sarkar[2,†]

[1]Jadavpur University
[2]Jadavpur University

Abstract

In this work, we present a simple and effective method to automatically generate highlights from scientific papers or their abstracts. The goal is to identify the most important sentences in an abstract that best represent its key ideas. We use two different models, one is the XGBoost Regressor Model and the XGBoost Classifier Model. In the XGBoost Classifier Model, each sentence is determined by comparing it with the reference highlight; if the sentence overlaps significantly with the highlight, it is assigned 1; otherwise, it is assigned 0. The XGBoost Regressor Model captures the gradient of importance, allowing finer ranking to score each sentence based on how similar it is to the human-written highlights (like 0.4, 0.9, 0.6, etc.). The models learn from these scores and select the top sentences to produce highlights. We tested our methods on training, validation, and test datasets. The results show that the regressor model performs better than the classifier model in most cases, especially when using ROUGE-1, ROUGE-2, and ROUGE-L evaluation metrics. In this shared task, out of many participating teams, 12 were selected, and our team secured the 8th rank. However, the difference between our XGBoost Regressor model's result and that of the top-ranked team is very small. Our approach is easy to apply and can be useful for automatic summarization tasks in scientific writing.

Keywords

Highlight Generation, Scientific Papers, XGBoost, Regressor, Classifier

1. Introduction

Scientific research papers often include a short summary section called "highlights" that captures the main points of the work. These highlights help readers quickly understand the key contributions of a paper. However, writing highlights manually can be time-consuming for authors, especially when dealing with many documents. This has created a need for automatic methods to generate highlights from scientific texts.

In this work, we focus on generating highlights from the abstract section of scientific papers. Abstracts are already short summaries, but they often contain multiple sentences that vary in importance. Our goal is to identify the most important sentences from the abstract and use them as highlights.

We explore machine learning techniques for this task, comparing classification and regression-based models. Our experiments show that the regression-based XGBoost model outperforms the classifier, offering finer-grained and more accurate sentence ranking. The model is trained using sentence-level features and uses ROUGE-based similarity scores as supervision.

This paper presents a simple, fast, and effective pipeline for sentence-level highlight prediction. We evaluate our approach using standard metrics and test it on separate datasets to confirm its general

Forum for Information Retrieval Evaluation, December 17-20, 2025, Varanasi, India
*Corresponding author.
†These authors contributed equally.
roypapun.md@gmail.com (P. Roy); jukamal2001@yahoo.com (K. Sarkar)
0009-0006-6140-5695 (P. Roy); 0000-0002-0689-3976 (K. Sarkar)

performance.

Our main contributions are:

1. We build a sentence-level highlight prediction system using XGBoost Regressor as well as Classifier.
2. We use simple string overlapping (for the Classifier) and ROUGE-L similarity (for the Regressor) scores between sentences and reference highlights as training labels.
3. We compare regression and classification approaches and show that regression performs better.
4. We provide a complete pipeline that includes training, validation, and testing.

2. State of the Art

Automatic summarization is a well-studied problem in natural language processing (NLP). Traditionally, summarization methods are grouped into two types: extractive and abstractive. Extractive methods aim to select important sentences from the original text, while abstractive methods generate new sentences based on the meaning of the text [1].

For extractive summarization, early approaches used statistical features such as term frequency (TF), inverse document frequency (IDF), sentence position, and similarity measures. Classical machine learning models like Naive Bayes, SVMs, and decision trees were later applied using these features [2].

In recent years, deep learning models such as BERT and its variants (e.g., BERTSum, T5) have shown strong performance in summarization tasks. These models use pre-trained language representations to understand sentence semantics and context [3]. However, they often require large datasets, long training times, and significant computing resources.

To reduce complexity and resource usage, some researchers have explored tree-based models like XGBoost for sentence scoring and selection. While classification models are often used to decide whether a sentence is a highlight or not, regression models provide a more fine-grained ranking of sentence importance [4]. However, there is limited research that directly compares these two strategies.

The task of generating research highlights from scientific texts lies at the intersection of *automatic summarization* and *scientific document processing*. Early work in this field focused mainly on extractive methods, where sentences from the source document are selected to form a summary. Such approaches were often based on statistical or heuristic features like sentence position, frequency, or TF-IDF weighting. While simple, these approaches struggled with semantic understanding and often produced highlights that were redundant or lacked coherence.

With the advent of *deep learning*, research in highlight generation has moved towards more sophisticated models. For instance, Rehman et al. [5] proposed the use of *sequence-to-sequence models with attention, pointer-generator networks, and coverage mechanisms* to generate highlights directly from abstracts, achieving promising ROUGE and METEOR scores on a large dataset of scientific papers[5]. Their work demonstrated that abstractive methods can better capture the essence of a paper compared to purely extractive approaches.

Later advancements introduced *domain-specific embeddings* to further enhance highlight generation. Rehman et al.[6] combined *pointer-generator networks with SciBERT embeddings*, trained specifically on scientific texts, to generate higher-quality highlights. Their system outperformed baseline models on benchmark datasets such as CSPubSum and MixSub, achieving state-of-the-art results with significantly improved ROUGE, METEOR, and BERTScore metrics [7].

In subsequent work, Rehman and colleagues explored multiple strategies to improve scientific summarization. They introduced contextual embedding-based models using ELMo [6], and proposed a named-entity-driven highlight generation approach using NER features [8]. Further, they analysed abstractive text summarization techniques leveraging pre-trained transformer models [9] and provided an extensive study on highlight generation from abstracts in EEKE [5]. Together, these studies represent the most comprehensive research effort in automatic highlight generation from scientific texts, showing how deep contextual models and hybrid architectures can enhance both factual accuracy and content relevance.

Parallel to this, the broader NLP community has witnessed groundbreaking improvements with *transformer-based pre-trained models*. Notably, BERT (Bidirectional Encoder Representations from Transformers) introduced by Devlin et al.[3] enabled the learning of deep bidirectional contextual embeddings, setting new benchmarks across multiple NLP tasks including summarization [3]. Subsequent models such as SciBERT and PEGASUS further specialized this pre-training for scientific or summarization-specific tasks.

Recent studies also highlight the limitations of abstractive methods, such as hallucinations and factual inconsistencies. To address these, hybrid approaches combining extractive and abstractive methods have been explored. For example, some works adopt extractive steps for sentence selection followed by abstractive refinement, ensuring both factual correctness and conciseness.

Despite these advances, *most state-of-the-art systems rely on large-scale pre-trained models requiring extensive computational resources.* This poses challenges for practical deployment. In contrast, our work focuses on a more lightweight yet effective approach: using *XGBoost regression and classification models* with sentence-level features to predict highlights. By comparing extractive strategies under a machine learning framework, we aim to provide a resource-efficient alternative while still maintaining competitive performance.

Our work fills this gap by providing a direct comparison between regression and classification models for extractive summarization, using XGBoost. We show that the regression approach performs better, especially when evaluated with ROUGE metrics.

3. Methodology

Our method is based on extractive summarization at the sentence level. The main idea is to rank sentences from an abstract based on how well they match the human-written highlights.

The goal of our research is to find an effective and light- weight method to automatically generate sentence-level highlights from scientific abstracts. Specifically, our goal is to compare two different machine learning strategies, classification and regression, using the XGBoost model. Our main motivation of this research is can we find or collect the highlights from a scientific paper? For that we use a predefined model named "XGBoost" [4]. Here we use two different part of the same model one is classification model and another is regression model and here we see which approach, classification or regression using XGBoost, performs better in extracting meaningful highlights from abstracts.

To answer this question, we designed an experiment where both XGBoost Classifier and Regressor are trained and tested on the same dataset, using the same feature representations and evaluation metrics.

Now, we describe our methodology in detail, corresponding workflow diagram to the steps shown in the Figure 1. Each step is critical for achieving the overall goal of generating sentence-level highlights from abstracts using XGBoost.

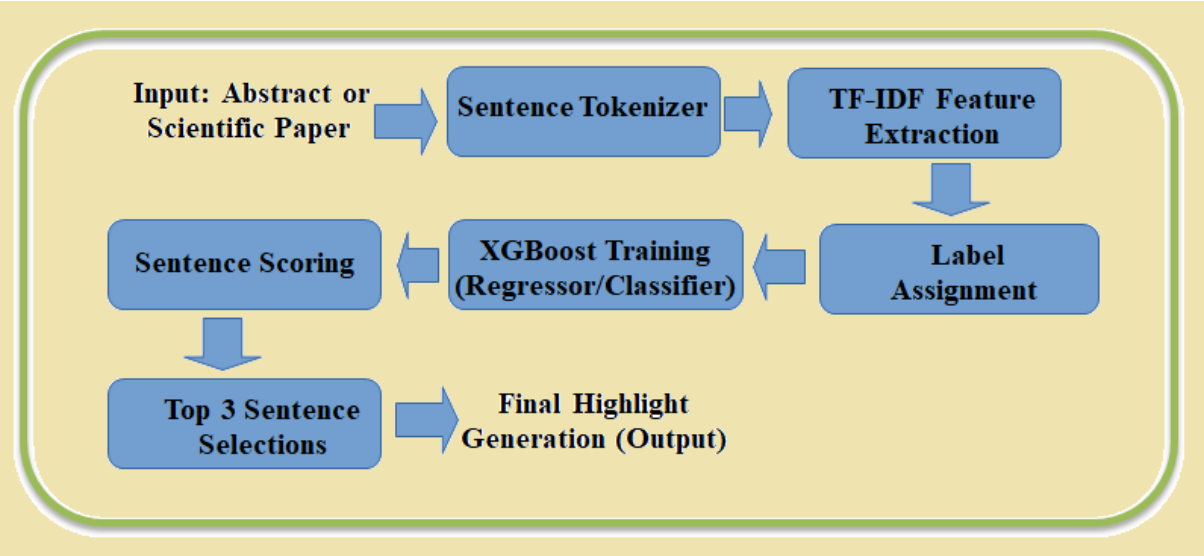

Figure 1: Flow of the Experiment

The steps in our method are as follows:

3.1. Input – Abstract or Scientific Paper:

Generally in this process, we begin with the input, which is typically a scientific abstract or, in some cases, the full text of a research paper. The abstract is chosen as the primary input because it provides a condensed version of the entire work and often contains the essential contributions, background, and findings of the study. By focusing on abstracts, we reduce the complexity of the summarization task while still capturing meaningful highlights. The research question guiding this step is: *Can we extract highlights directly from abstracts in a way that mirrors human-written summaries?* Preparing the input involves ensuring that the text is clean, free of unnecessary formatting, and ready for processing. This preparation phase ensures consistency across datasets and provides a uniform starting point for subsequent steps.

Here we used the well-organized data set given by the *FIRE-2025* organizer, which have 3 different fields like Filename, Abstract, and Highlights.

We used three data sets: training, validation, and testing. Each entry contains a filename, an abstract, and (except in the test data set) the ground-truth highlights.

3.2. Sentence Tokenization:

Once the abstract is obtained, the first technical step is sentence tokenization. This process splits the abstract into individual sentences. We use the Natural Language Toolkit (NLTK) tokenizer[10] , which is well-suited for handling scientific text. Tokenization is essential because the model operates at the sentence level, evaluating each sentence separately to determine its likelihood of being part of the highlight. The accuracy of tokenization is critical: poorly segmented sentences could lead to broken meaning units, reducing the quality of extracted features and final predictions. Beyond simple splitting, sentence tokenization prepares the foundation for aligning sentences with their corresponding highlight scores, which is a key element of supervised training in both classification and regression

approaches.

3.3. TF-IDF Feature Extraction:

After tokenization, we transform each sentence into a numerical representation using Term Frequency–Inverse Document Frequency (TF-IDF)[11] [12]. This step captures the importance of words in a sentence relative to the entire dataset. Sentences that contain domain-specific keywords or frequently occurring technical terms often have higher TF-IDF weights, making them more likely candidates for highlights. TF-IDF is advantageous because it balances local word importance (term frequency within a sentence) with global rarity (inverse frequency across the corpus). This transformation results in feature vectors that XGBoost models can interpret. While deep embedding models like BERT capture context-rich features, TF-IDF is computationally efficient, interpretable, and sufficient for our lightweight summarization framework. TF-IDF is used to convert each sentence into numerical features, capturing word-level importance. Features are normalized using Min-Max scaling.

3.4. Label Assignment:

Label generation is the core step for supervised training. For the classifier, we assign binary labels: a sentence is labeled **1** if it overlaps significantly with the reference highlight, and **0** otherwise. For the regressor, we compute a continuous similarity score using ROUGE-L between each sentence and the reference highlight. For example, sentences receive scores such as 0.4, 0.9, or 0.6 depending on their similarity. This approach allows the regression model to capture nuanced differences in sentence importance. Label assignment ensures that the model has reliable ground truth data to learn from. The careful design of labels is critical, since poor labeling could mislead the model into learning irrelevant sentence patterns. This step effectively aligns sentences with their "highlight value," setting the stage for effective model training. For the regression model, each sentence is assigned a score based on its ROUGE-L similarity with the reference highlight [10].

3.5. XGBoost Training (Regressor/Classifier):

In this step, we train two separate models: the XGBoost Classifier and the XGBoost Regressor. XGBoost (Extreme Gradient Boosting) [4] is chosen for its efficiency, ability to handle sparse TF-IDF vectors, and strong performance in many text-related tasks. XGBoost builds many small decision trees, each fixing the errors of the previous one, guided by gradient information. With regularization, it balances accuracy and generalization. A model is required that performs efficiently with TF-IDF features (sparse, high-dimensional), Transformers (like BERT) require huge resources; XGBoost is lightweight and still effective. Instead of building one strong model, XGBoost builds an ensemble of weak learners (decision trees). Each new tree tries to correct the errors made by the previous trees. Over many iterations, the ensemble becomes very strong.

1. **XGBoost Classifier Model**
   The XGBoost Classifier is a supervised machine learning algorithm based on the principle of gradient boosted decision trees. It is designed for binary or multi-class classification tasks and is known for its high efficiency, scalability, and predictive performance. In the context of highlight generation, the XGBoost Classifier is used to determine whether a sentence from an abstract should be included in the final highlight or not.
   At its core, XGBoost builds an ensemble of weak learners (decision trees) in a sequential manner, where each new tree attempts to correct the errors made by the ensemble of previous trees. The classifier optimizes a logistic loss function through gradient boosting, where the gradients (first derivatives) and hessians (second derivatives) of the loss function guide how new trees are constructed. This enables the model to efficiently capture non-linear relationships in the data.

In our implementation, sentences are first transformed into TF–IDF feature vectors, which represent the importance of words within the abstract relative to the corpus. Each sentence is then assigned a binary label:

- 1 if it matches or overlaps with the reference highlight (indicating it should be selected),
- 0 otherwise.

The XGBoost Classifier then learns to distinguish between these two classes. During training, the model minimizes the binary cross-entropy loss, which measures the divergence between the predicted probability and the true label. The final model outputs a probability score between 0 and 1 for each sentence, reflecting the likelihood of that sentence being part of the highlight. Sentences with probabilities above a certain threshold (commonly 0.5) are classified as highlights. An important feature of the XGBoost Classifier is its regularization mechanism, which penalizes overly complex trees and prevents overfitting. Additionally, XGBoost efficiently handles sparse input data, making it well-suited for TF–IDF representations where many entries are zero.
Overall, the XGBoost Classifier provides a fast, interpretable, and resource-efficient method for sentence-level highlight prediction. While it simplifies the task into a binary decision-making process, its performance demonstrates the effectiveness of boosted decision trees in scientific text summarization tasks.

2. **XGBoost Regressor Model**
The XGBoost Regressor is a variant of the gradient boosting framework designed for continuous prediction tasks. Instead of predicting discrete class labels, the regressor learns to assign a continuous score that reflects the degree of relevance or similarity. In the context of highlight generation, this makes the XGBoost Regressor particularly suitable, since highlight-worthy sentences are not always strictly binary (highlight or non-highlight), but may lie on a spectrum of importance.
Similar to the classifier, the regressor builds an ensemble of decision trees sequentially, where each new tree corrects the residual errors of the previous trees. However, instead of optimizing logistic loss, the XGBoost Regressor minimizes the mean squared error (MSE) between the predicted score and the target value. This design allows the model to learn subtle differences in sentence importance, producing finer-grained predictions compared to the binary framework of the classifier.
In our system, sentences from abstracts are first transformed into TF–IDF vectors, capturing the importance of words in relation to the overall corpus. Each sentence is then assigned a continuous label based on its similarity to the reference highlight. For example, if a sentence has strong lexical overlap with the reference highlight, it receives a higher score (e.g., 0.9), while a less relevant sentence may receive a lower score (e.g., 0.2). These continuous labels serve as the ground truth for training the regressor.
During training, the regressor learns to map TF–IDF features to similarity scores, guided by the gradients and hessians of the MSE loss. The output of the model is a real-valued score for each sentence, indicating how closely it resembles the human-written highlight. Sentences are then ranked by these predicted scores, and the top-k (in our case, top 3) are selected as the generated highlights.
The XGBoost Regressor also benefits from regularization mechanisms (L1 and L2 penalties) and efficient handling of sparse TF–IDF vectors. These features allow it to maintain a balance between accuracy and generalization while remaining computationally efficient.
Overall, the XGBoost Regressor offers a ranking-based approach to highlight generation. By capturing graded relevance rather than enforcing strict binary decisions, it provides more flexible and accurate sentence selection. This explains why, in our experiments, the regressor consistently outperformed the classifier, achieving higher ROUGE scores and better alignment with the reference highlights.

The classifier learns to distinguish between highlight and non-highlight sentences, while the regressor learns a continuous mapping from sentence features to highlight scores. One of the key contributions of our work is the comparison of these two approaches head-to-head. We found that regression provides finer granularity in ranking, whereas classification is limited to a binary decision boundary.

3.6. Sentence Scoring and Ranking:

Once trained, the model is used to assign scores to each sentence in unseen abstracts. In the case of the classifier, this is a probability score indicating the likelihood of being part of the highlight. For the regressor, this is a continuous value reflecting how closely the sentence matches the ground truth highlight. Sentence scoring is where the model applies its learned knowledge, and the quality of these scores directly determines the effectiveness of the final highlight generation. By ranking sentences according to these scores, the system ensures that the most relevant and informative sentences are prioritized.

3.7. Sentence Selection:

After scoring, the next step is to select the top three sentences with the highest scores. This number was chosen based on the common practice of research journals requiring three highlights. Selecting three sentences strikes a balance between conciseness and coverage, ensuring that the highlights provide a quick yet informative overview of the abstract. The ranking process is deterministic: sentences are sorted in descending order of their scores, and the top three are chosen. This step operationalizes the sentence-level scoring into a usable summary format. For unseen abstracts (for test dataset abstracts), the trained model predicts a score for each sentence. We rank the sentences and select the top 3 as the predicted highlight.

3.8. Final Highlight Generation (Output):

The last step produces the final highlight set for each abstract. The selected top three sentences are presented together as the predicted highlight, which can then be compared against the human-written highlight for evaluation. Evaluation is conducted using ROUGE-1, ROUGE-2, ROUGE-L[10], and sentence-level accuracy. This step closes the loop of the workflow, translating the model's predictions into a concrete, human-readable output. The generated highlights can then be used in academic repositories, search engines, or research support tools to help readers quickly grasp the essence of a scientific paper.

By elaborating on each step, we demonstrate how the workflow systematically transforms raw text into structured highlights. Each component contributes uniquely to the final outcome, and together, they provide a coherent pipeline for extractive highlight generation using XGBoost.

4. Dataset Details

The dataset used in this study was provided as part of the SciHigh shared task at FIRE 2025 (Forum for Information Retrieval Evaluation). It contains three subsets: training, validation, and test data. Each record in the dataset includes the paper filename, abstract, and human-written highlights (except in the test set). The dataset is derived from the MixSub corpus, which includes research papers from multiple scientific domains. Following standard practice, we use the training and validation data for model development and report test performance based on the leaderboard results. Details of the dataset and the shared task can be found in the official FIRE 2025 SciHigh track documentation [7].

5. Results

We conducted extensive experiments using both XGBoost Classifier and XGBoost Regressor models for the task of highlight generation. The evaluation was performed using widely accepted metrics: ROUGE-1, ROUGE-2, ROUGE-L, and sentence-level accuracy.

From our experiments, the regressor consistently outperformed the classifier. This is because the regression framework allows the model to capture partial relevance: a sentence may not be an exact match to the reference highlight but can still be closer than others. By assigning continuous values, the regressor provides a finer ranking of sentences, which is more suitable for extractive highlight generation than the rigid binary decisions of the classifier. Initially, our validation experiments indicated that the regression model (run1) provided better performance across all evaluation metrics compared to the classification model. The validation results are summarized in Table 1.

Table 1
Performance comparison of Classifier and Regressor models

Metric	Classifier	Regressor (run1)
ROUGE-1 (F1)	0.3728	**0.3981**
ROUGE-2 (F1)	0.1324	**0.1557**
ROUGE-L (F1)	0.3741	**0.3935**
Accuracy	0.65	**0.71**

Following these experiments, we submitted the outputs of our XGBoost Regressor model to the evaluation server for final benchmarking against other participating teams. Here we show the full leaderboard of the 12 selected teams result in Table 2.

Table 2
All Participant Performance Table

Group Name	Run Submission (Best Run Taken)	ROUGE-L	Rank
Text_highlights_gen	run1	0.2345	1
AiNauts	run1	0.2324	2
SVNIT_CSE	run1	0.2302	3
NLPFusion	**run2**	**0.2296**	**4**
The NLP Explorers	run2	0.2294	5
NIT_PATNA_2025	run1	0.2242	6
MUCS	run1	0.2208	7
JU_CSE_PR_KS	**run1**	**0.2206**	**8**
SCaLAR	run1	0.2033	9
Ayanika	run1	0.1791	10
Shilpo	run1	0.1675	11
TJP	run1	0.0673	12

In this table our team name, identified as *JU_CSE_PR_KS*, the regressor model (*run1*) achieved a ROUGE-L score of **0.2206** on the official test set. In the leaderboard, our team ranked *8th overall*, which demonstrates the competitiveness of our approach despite its simplicity.

This result confirms that while deep learning models (such as BERT-based architectures) often dominate leaderboards, carefully designed lightweight models like XGBoost can still achieve strong performance in automatic highlight generation. Our method, relying solely on TF-IDF features and gradient-boosted decision trees, provides a computationally efficient alternative to transformer-based models, making it more suitable for resource-constrained environments.

Moreover, achieving 8th place out of multiple participating systems validates the effectiveness of

using regression-based scoring rather than binary classification for extractive summarization. The regressor's ability to assign continuous importance values allows for better ranking of candidate sentences, leading to improved overlap with human-written highlights.

Overall, the results highlight the trade-off between efficiency and absolute performance. Although our ROUGE-L score (0.2206) is definitely lower compared to top transformer-based systems, but the margin is very small (**0.009** from the fourth team). Our approach is interpretable, faster to train, and significantly less resource-intensive. These qualities make the model practical for real-world deployment in digital libraries, academic repositories, and summarization tools where scalability and speed are as important as accuracy.

Here we give an model result comparison chart which will explain well our result shows in Figure-2

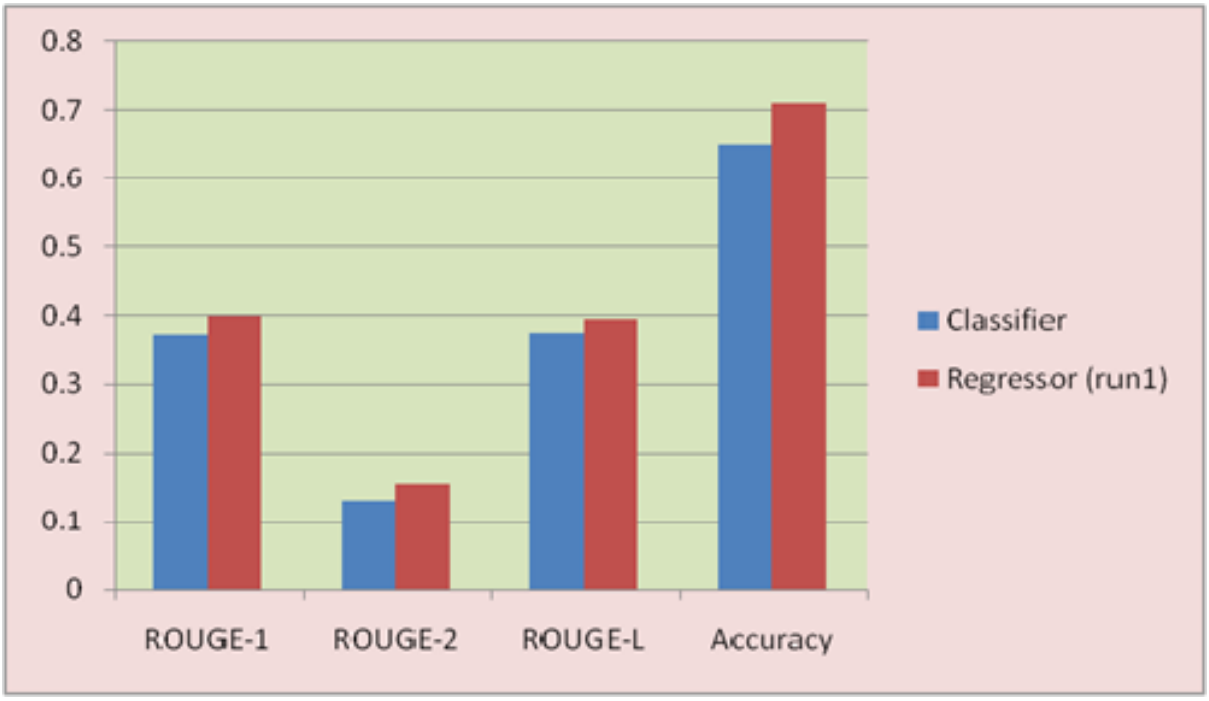

Figure 2: comparing the model results XGBoost Classifier vs Regressor

The regressor model (run1) provided better ranking of important sentences and achieved higher overlap with human-written highlights. The final model was used to generate highlights for the test dataset, and the results were saved as a CSV file. This demonstrates that effective extractive summarization can still be achieved without relying on large-scale pre-trained transformers.

6. Discussion

The experimental findings and the results of the leader board provide valuable insight into the strengths and limitations of our approach. One of the most important observations is that the regression model consistently outperforms the classifier, we see in local validation and in the official evaluation we can see that our regressor model works well. This confirms our hypothesis that assigning continuous importance scores to sentences allows the model to better capture nuances in sentence relevance than binary classification. By treating highlight prediction as a ranking problem rather than a simple yes/no decision, we enable the system to prioritize sentences with subtle but meaningful differences in importance.

Another important aspect of the discussion is the trade-off between lightweight machine learning models and resource-intensive transformer-based models. While deep learning approaches such as

BERT, SciBERT, and PEGASUS dominate in absolute performance on summarization tasks, they require significant computational power, large annotated datasets, and considerable training time. In contrast, our XGBoost-based system relies on TF-IDF features and gradient-boosted decision trees, making it interpretable, easy to train, and computationally efficient. This makes our approach especially suitable for institutions with limited resources or for deployment at scale in digital libraries and repositories where speed and efficiency are crucial.

The official leader board result, where our team JU_CSE_ PR_KS achieved 8th place with a ROUGE-L score of 0.2206, further demonstrates the practical competitiveness of our model. Despite being narrowly behind the top-ranked team, which achieved a ROUGE-L of 0.2345, the difference is relatively small. This indicates that carefully engineered classical machine learning models can still remain competitive against modern deep learning systems in certain contexts. Furthermore, our validation results, which showed ROUGE-L values above 0.39, suggest that the model generalizes reasonably well even though performance drops on unseen test data, a common issue in NLP tasks.

A limitation of our current system is its reliance solely on surface-level lexical features, which means it may miss deeper semantic relationships between sentences and highlights. This explains why transformer-based models, which encode context and semantics more effectively, achieve slightly higher scores. Future work could focus on hybrid models that integrate semantic embeddings (such as sentence transformers) into the XGBoost framework, thus combining efficiency with improved representation.

Overall, the discussion highlights that our regression-based XGBoost approach provides a good balance between accuracy, interpretability, and efficiency. It achieves competitive results on a challenging benchmark task, validates the usefulness of regression for sentence-level scoring, and opens pathways for future research into hybrid or feature-augmented systems that bridge the gap with large-scale deep learning models.

7. Conclusion

In this work, we investigated the task of automatic highlight generation from scientific abstracts using XGBoost-based models. Our research focused on comparing two approaches: classification and regression. Through systematic experimentation, we demonstrated that the regression-based model provides more effective sentence ranking and generates highlights that better align with human-written ones. Validation experiments confirmed that the regressor consistently outperforms the classifier in terms of ROUGE scores and accuracy, and the official leader board evaluation placed our team, JU_CSE_PR_KS, in 8th position with a ROUGE-L score of 0.2206.

The main contribution of our study lies in showing that even lightweight, interpretable models like XGBoost can achieve competitive performance against more resource-demanding transformer-based systems. This makes our approach attractive for environments with limited computational resources and for large-scale applications where efficiency and interpretability are crucial. By leveraging TF-IDF features, sentence tokenization, and regression-based scoring, we designed a workflow that is not only effective but also practical to implement and deploy.

However, the study also highlights certain limitations. The reliance on surface-level lexical features restricts the model's ability to capture deeper semantic relationships between sentences and highlights. While this keeps the system lightweight, it also explains why transformer-based systems can achieve slightly higher performance. A promising direction for future work is the integration of semantic embeddings, such as BERT or SBERT [3] or sentence-transformer vectors, with XGBoost to create a hybrid system that maintains efficiency while improving semantic understanding.

Declaration on Generative AI

The authors declare that generative AI tools, such as ChatGPT, were used to assist in language editing, grammar checking, and formatting during manuscript preparation. All intellectual contributions, analyses, and experimental results are solely the authors' own work.

References

[1] C.-Y. Lin, Rouge: A package for automatic evaluation of summaries, in: Text summarization branches out, 2004, pp. 74–81.

[2] R. Nallapati, F. Zhai, B. Zhou, Summarunner: A recurrent neural network based sequence model for extractive summarization of documents, in: Proceedings of the AAAI conference on artificial intelligence, volume 31, 2017.

[3] J. Devlin, M.-W. Chang, K. Lee, K. Toutanova, Bert: Pre-training of deep bidirectional transformers for language understanding, in: Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers), 2019, pp. 4171–4186.

[4] T. Chen, C. Guestrin, Xgboost: A scalable tree boosting system, in: Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining, 2016, pp. 785–794.

[5] T. Rehman, D. K. Sanyal, S. Chattopadhyay, P. K. Bhowmick, P. P. Das, Automatic generation of research highlights from scientific, in: 2nd Workshop on Extraction and Evaluation of Knowledge Entities from Scientific Documents (EEKE'21), collocated with JCDL'21, 2021.

[6] T. Rehman, D. K. Sanyal, S. Chattopadhyay, Research highlight generation with elmo contextual embeddings, Scalable Computing: Practice and Experience 24 (2023) 181–190.

[7] T. Rehman, D. K. Sanyal, S. Chattopadhyay, P. K. Bhowmick, P. P. Das, Generation of highlights from research papers using pointer-generator networks and scibert embeddings, IEEE Access 11 (2023) 91358–91374. doi:10.1109/ACCESS.2023.3292300.

[8] T. Rehman, D. K. Sanyal, P. Majumder, S. Chattopadhyay, Named entity recognition based automatic generation of research highlights, in: Proceedings of the Third Workshop on Scholarly Document Processing (SDP 2022) collocated with COLING 2022, Association for Computational Linguistics, Gyeongju, Republic of Korea, 2022, pp. 163–169. URL: https://aclanthology.org/2022.sdp-1.18.

[9] T. Rehman, S. Das, D. K. Sanyal, S. Chattopadhyay, An analysis of abstractive text summarization using pre-trained models, in: Proceedings of International Conference on Computational Intelligence, Data Science and Cloud Computing: IEM-ICDC 2021, Springer, 2022, pp. 253–264.

[10] S. Bird, E. Klein, E. Loper, Natural language processing with Python: analyzing text with the natural language toolkit, " O'Reilly Media, Inc.", 2009.

[11] K. Sparck Jones, A statistical interpretation of term specificity and its application in retrieval, Journal of documentation 28 (1972) 11–21.

[12] A. F. AlShammari, Implementation of keyword extraction using term frequency-inverse document frequency (tf-idf) in python, Int. J. Comput. Appl 185 (2023) 9–14.