CHAPTER 1

INTRODUCTION

1.1 GCN-BERT APPROACH

The GCN-BERT approach represents a ground breaking fusion of Graph Convolutional Networks (GCNs) and Bidirectional Encoder Representations from Transformers (BERT), specifically designed to address the unique challenges of mathematical information retrieval (MIR). Traditional retrieval systems often struggle with mathematical content due to its two-dimensional structure, symbolic nature, and context-dependent semantics. By combining GCNs' ability to model structural relationships in mathematical expressions with BERT's strength in understanding natural language context, our hybrid approach achieves superior performance in retrieving and ranking scientific documents containing both formulas and text. Mathematical expressions are inherently graph-structured data, where symbols (nodes) are connected via operators or spatial relationships (edges). The textual context surrounding formulas (e.g., definitions, explanations) is processed using BERT to generate context-aware embeddings that disambiguate symbols (e.g., "E" as energy in physics vs. expectation in statistics). Capture domain-specific terminology and conceptual linkages between formulas and their descriptions. Support cross-modal alignment, where text descriptions inform formula interpretation and vice versa.

1.2 MOTIVATION FOR THE PROJECT

The exponential growth of scientific literature has made the retrieval of documents containing mathematical expressions a critical challenge.

Mathematical formulas are exact, concise expressions of sophisticated concepts, but their two-dimensional nature and variability in expression (e.g., $a^2 + b^2 = c^2$ vs. $c^2 = a^2 + b^2$) present special challenges to conventional search systems. Existing keyword-based methods do not preserve the semantic equivalence between syntactically distinct formulas or with the explanatory text surrounding them. This limitation significantly hinders effective scientific information retrieval, particularly in fields like physics, engineering, and mathematics where formulas are ubiquitous. The need for systems that can simultaneously understand formula structure and contextual meaning motivates our research into hybrid neural approaches.

1.3 OBJECTIVE AND SCOPE

The primary objective of this research work is to develop a robust mathematical information retrieval system that leverages the complementary strengths of GCNs and BERT. This research focuses on the retrieval of documents containing mathematical expressions paired with explanatory text, such as academic papers, educational resources, and technical reports. It includes the following goals.

The model is trained and tested on the IM2Latex dataset, which comprises 20,00,00 mathematical formulas represented in latex form, which is suitable for this project.

1) To Use of GCNs to represent mathematical formulas into graphs, where nodes denote either symbols or operators and edges depict their hierarchical relations. This in turn would let models recognize their semantically equivalent expressions regardless of syntactic differences.

2) To Use BERT to process all the textual descriptions about the formulas to capture field-defining, meaning-making explanations and conceptual family ties that shed light on the formulas' goals.

3) To Create a weighted similarity scoring scheme to integrate GCN-based Formula embeddings and BERT-based Text embeddings to ensure just consideration between the two Modalities during Retrieval.

4) Evaluating the model on the dataset using standard metrics such as normalized Discounted Cumulative Gain (nDCG) and mean Average Precision (mAP) and benchmarking against currently existing methods to show how it outperforms.

1.4 ORGANIZATION OF THE REPORT

Chapter 1: Introduction -This chapter introduces the motivation and problem definition of the project in detail of GCN and BERT approach.

Chapter 2: Literature Survey-This chapter provides an overview of the reference documents collected for this project, highlighting key takeaways and existing methodologies in math expression retrieval using GCN method and BERT based transformers.

Chapter 3: Project Description-In this chapter, the project's description is provided, including a discussion of existing work in the field, the proposed methodology, and the benefits of the project.

Chapter 4: Architecture Design -This chapter presents the architecture of the proposed solution, offering a comprehensive explanation of the math retrieval GCN model and evaluation of similarity score.

Chapter 5: Software and Hardware Requirements - Here, the software and hardware requirements needed to execute the project are specified, along with the technologies used.

Chapter 6: Module Description-This chapter describes the various modules within the project,

Chapter 7: Implementation-In this chapter, the complete implementation process of the project is discussed, detailing the steps taken to realize the proposed methodology.

Chapter 8: Results and Analysis - This chapter gives the results of the project, along with the explanations of the results of each module, supplemented by graphical representations.

Chapter 9: Conclusion and Future work - This chapter brings the project to a conclusion, outlining the findings and possibilities for future extension and implementation.

CHAPTER 2

LITERATURE REVIEW

This chapter examines the applicable works of a number of researchers, along with their methodologies, performance, and limitations. In 2023, Hyesoo Kong et al. [1] proposed a novel approach to automatic metadata extraction for Korean academic papers using a BERT-based framework. The researchers addressed the challenges posed by unstructured PDFs and limited Korean training data by constructing the largest labeled corpus to date, comprising 315,320 papers from 503 academic journals. The team developed two advanced models: KorSciBERT-ME-J, which incorporates journal format information, and KorSciBERT-ME-J+C, which additionally integrates coordinate information.

Transfer learning to enhance the metadata extraction has been done by both models BERT based. Their method involved PDF processing to lay the boxes and extract features such as text, coordinates, and font sizes. A rule-based automatic labeling method was developed and doing rigorous data inspection to deliver reliability by the authors. KorSciBERT-ME-J is noted to have attained the highest F1-score of 99.36%, showing sound performance across several fields of metadata. This indicates the proficiency of pre-trained language models and specialist metadata annotations in scalable and high-accuracy metadata extraction from complex academic documents.

This was in the year 2023, whereby Kim et al. [2] introduced a new way of extracting alcohol-related information from unstructured bilingual clinical notes using transformer-based models. Leveraging their application for Korean and English languages, the authors employed a multilingual transformer-XLM-RoBERTa. Minimal preprocessing was done, whereby symbols were changed and acronyms extended without the aid of outside ontologies. The results showcase how the framework distinctly made differences in alcohol-related information from that of typical clinical data, achieving a macro F1 score of 84.70%. In addition, the study presented the robustness of this model in the bilingual clinical context, further emphasizing the relevance of linguistic contextualization in the multilingual EMR environment.

In 2023, as indicated by Hambarde et al. [3], has done a massive review of information retrieval techniques, with special focus on advances in deep learning and NLP. The study described retrieval in two stages and applied three transformer-based models, viz., BERT and GPT-2, to enhance retrieval tasks for the first stage and second stage, respectively. It outlines the problems with classical term-based models, along with techniques such as semantic embeddings, multi-modal retrieval, and knowledge integration through external sources like knowledge graphs. Their work emphasized the need for dense and hybrid retrieval approaches; it provided an exhaustive discussion of neural and lexical models for document retrieval fine-tuning.

Improvements were made in a transformer-based NLP model by Sushant Singh et al. in the year 2017. Adding the attention mechanism was the main thrust of this advancement for enhancing the processing of long sequences. Their work delved into diverse paths, including transfer learning where pretrained models can reduce training time and generalize better. They also used knowledge distillation to build efficient yet high-performing models and used pruning for redundant model-parameter reduction to lessen complexity. Also, the researchers relied on quantization to lessen model size without affecting accuracy. All these techniques, put together, were aimed at finding the best ground in model accuracy versus resource consumption.

In 2020, Xinyun Cheng et al. [5] improved upon a unique combined method utilizing multiple Natural Language Processing (NLP) libraries for software document analysis with better accuracy. The authors recognized the individual limitations of NLP libraries and proposed a two-step process of document-level library selection and sentence-level overwriting. The document-level approach assigns a majority NLP library based on inter-library output overlap. For cases of less overlap, result refinement actively overwrites default results with other libraries that offer specific improvements.

The work was evaluated using four popular NLP libraries: NLTK, spaCy, Stanford CoreNLP, and OpenNLP on 200 software documents, which were all different from each other and touching different tasks like tokenization and part-of-speech tagging. Results show that this combined mode has given better accuracy in the above tasks, around 2% increase over any standalone library. This demonstrates the efficacy of the method in availing individual strengths of each library for very accurate software document analysis. 4 An analysis done by Jabbar et al. in 2023[6] examined stemming techniques with a different perspective of information retrieval (IR) and NLP applications. The review categorizes stemming methodologies into linguistic, statistical, corpus-based, context-sensitive, and hybrid approaches, assessing their applicability across languages and specific NLP tasks. Key contributions include an analysis of stemming algorithms, an evaluation of performance metrics, and the identification of challenges in stemmer design. Their work provides a foundation for future research in stemming techniques, particularly for complex multilingual NLP applications.

In 2021, Houssein et al. [7] performed an in-depth review with a focus on machine learning solutions in biomedical NLP for the processing of electronic health records (EHR) analysis. The research highlights the combination of NLP and machine learning methods in handling the increasing amount of clinical data in EHRs, which generally hold unstructured clinical narratives. Their research shows the promise of these methods in clinical decision support, prediction of chronic diseases, and assessment of risk, with future directions and challenges in applying biomedical NLP for medical progress.

In 2022, Cherubin Mugisha et al. [8] presented a comparative study on NLP pipelines for outcome prediction using medical notes. The study compared traditional and transformer-based models like BERT, Bio BERT, and Clinical BERT, employing diverse preprocessing methods including NER-based filtering and contextual word embeddings. Using admission notes from the MIMIC-III database, the study achieved an F1-score of 98.2% for outcome prediction.

The results highlighted the effectiveness of advanced contextualized embeddings and emphasized the importance of thorough preprocessing. In 2022, Shunli Zhang et al. [9] proposed a Bi-LSTM-CRF neural network for clinical event extraction, emphasizing medical knowledge features. The model combined character level encoding via CNNs, word embeddings, and medical knowledge features extracted from clinical systems. This information was fed into a Bi-LSTM layer to capture context, followed by a Conditional Random Field (CRF) layer for label prediction. Evaluations on the THYME and 2012 i2b2 datasets demonstrated that their model outperformed existing rule-based, machine learning, and BERT-based methods, especially in recognizing low frequency clinical words. 5 In 2022, Lukesh Kadu et al. [10] conducted a comparative analysis of deep learning models for Twitter sentiment classification, evaluating the performance of several architectures, including Convolutional Neural Networks (CNN), Simple Recurrent Neural Networks (RNN), Long Short-Term Memory (LSTM), Bidirectional LSTM (Bi-LSTM), BERT, and RoBERTa. Through their work, it was discovered that that RoBERTa architecture had performed best among all architectures in terms of the accuracy of subsequent sentiment classification on Twitter data. The research explains the effectiveness of RoBERTa in handling complex sentiment data as compared to standard techniques of machine learning. K. S. Kalaivani et al. [11]

conducted research in the year 2022 on the performance of deep learning approaches in sentiment classification at document level. These researchers performed their comparisons on CNN, RNN, and LSTM models, concluding LSTM as providing the best accuracy due to its sequential information retention versus that of CNN and RNN. Also, this work tends to premise the applicability of LSTM in sentiment classification problems in which the order and contexts of words among texts are very important.

In 2019, Richa Sharma et al. introduced a hybrid deep learning model which seeks to find a solution to problems of unavailability of adequate linguistic resources and the morphological complexity of Hindi Named Entity Recognition (NER). The proposed model uses Bidirectional Long Short-Term Memory (Bi-LSTM), Conv-Nets, and Conditional Random Fields to effectively extract word- and character-level features. Word2Vec and GloVe pre-trained embedded representations have also been applied in this case to minimize the effect of meager labeled data and for character-level embeddings that assist better handling. The result of the experiment proved that the combined Bi-LSTM-CNN-CRF model performed better than baseline models by raising F-scores above those for RNN and LSTM. The approach captures context very well and successfully takes care of flexible word order in Hindi while improving accuracy in named entity recognition, and holds great potential for further evolving together with data.

CHAPTER 3

PROJECT DESCRIPTION

3.1 OVERVIEW

The project describes a hybrid model that combines Graph Convolutional Networks (GCNs) with BERT (Bidirectional Encoder Representations from Transformers) to attempt to radically transform mathematical information retrieval. Traditional search systems are troubled with mathematical content owing to its atypical two-dimensional definition and contextual dependency. The conceptual framework employed in this study provides a solution to these issues by using GCNs to understand the convoluted hierarchical relationships embedded in mathematical formulae by treating them as graphs in which symbols are characterized as nodes and operations define edges. Meanwhile, BERT operates on the natural language text surrounding the mathematical content to build semantic context, allowing the system to identify equivalent mathematical expressions that just differ in notation while discerning their meaning based upon the respective textual description.

The resulting model fusion of both platforms integrated with an innovative fusion mechanism and dynamically balanced structural features and textual features throughout the retrieval process. Trained on the IM2Latex200k dataset comprising mathematical problems and explanations, the evaluation of the proposed system shows far superior results against the existing methods with more than 0.9 nDCG@3 on formula-centric queries. The technology finds practical applications in scientific search engines and educational websites and digital libraries where formula-rich contents need to be retrieved with high accuracy. The project investigates the scalability optimization of the implemented system towards handling large repositories of documents, thus permitting possible future extensions into interdisciplinary areas such as physics, engineering, and chemistry. This represents a major step forward in making mathematical knowledge more easily accessible and searchable by integrating the understanding of formula structure and contextual language processing.

3.2 PROBLEM STATEMENT

The high growth rate of science literature now calls for efficient retrieval systems to handle documents that contain mathematical expressions. There are three fundamental problems that have complicated modern information retrieval technology to process mathematical content.

First, the structure of the notation in mathematics is quite different from that of natural languages. Formulas are two-dimensional and mainly incorporate very complex hierarchical relationships existing among operators and symbols. Such representation and deep analytics cannot be captured well by traditional text-based techniques. For example, something like (a + b)/c looks straightforwardly similar to a + b/c in linear text but is foo madly distinct mathematically. Current retrieval systems based on using statistical methods such as TF-IDF or keyword matching fail to capture these structural nuisances when a user queries mathematical concepts, providing incorrect answers most often.

Second, the number of manifestations a mathematical expression has is many - like, but not limited to, ½, 0.5, 1/2 - and they are extremely context sensitive. Disambiguating from them becomes tough since most symbol interpretations depend on context definitions-the definition of "E" may be energy in physics but expected value in statistics. Present structures fail to map formulae to contexts that explain them or identify semantic equivalence among diverse forms of expressing the same idea. This brings very much limitation to knowledge discovery in STEM where an accurate form of retrieval is vital in research and learning when working with dense formula content.

Such systems will be of great help to the researchers as well as educators and students in retrieving papers accurately, based on both mathematical relevance and contextual descriptions. This research provides a hybrid method that combines a deep understanding of language in context with structural understanding of scientific notation. Several such areas and applications of search engines, digital libraries, and educational platforms have a strong need for research.

3.3 PROPOSED SYSTEM

In this system, creating hybrid retrieval models for the proper handling of queries containing mathematical expressions and textual content combined. This is done by applying GCNs to analyze mathematical structures and BERT to analyze text.

3.3.1 BERT For Text Similarity

The model of language designed by Google, BERT, makes use of extremely large body of unlabeled text data for pre-training concerning deep bi-directional representations. BERT is context-adaptive embeddings as opposed to the traditional models such as word2vec, which are used with all the static word vectors regardless of their context. This effectively addresses the polysemy problem by demonstrating that one word can have numerous vector representations based on its context.

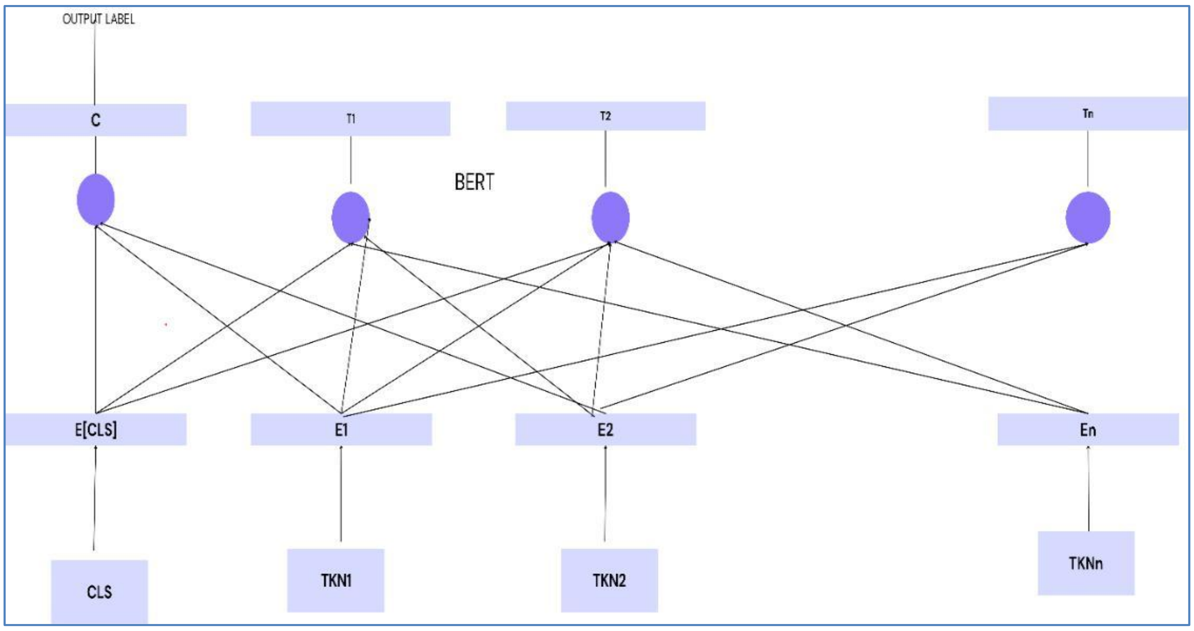


Figure:3.3.1 BERT Model

In Figure 3.3.1, the transformer model is quite close to the multilayer bidirectional transformer encoder that forms the basis of BERT's architecture. Self-attention in the encoder and attention in the decoder are used in this encoder-decoder network design. A pre-trained model can be specialized for a specific task by fine-tuning it using a smaller, task-specific dataset. Fine-tuning is the technique of adapting the pre-trained weights to fit the specific application more closely, as opposed to training from scratch.

i) **Text Preprocessing for BERT**

ET= BERT(Ti)

$$E_T = \text{softmax}\left(\frac{QK^T}{d_k}\right)V \qquad \text{----------(1)}$$

Where Q, K, and V represent the query, key, and value matrices, respectively,d_k is embedding dimension.

The SoftMax function normalizes the attention scores.

ii) Graph Representation of Mathematical Expressions

$$\boldsymbol{G_i} = (\boldsymbol{V_i}, \boldsymbol{E_i}) \qquad \text{----------(2)}$$

Where Vi represents the nodes (mathematical symbols). Ei represents the edges (relationships between symbols).

hv(0)=one-hot(v)

$$h_v^{(l+1)} = \sigma\left(\sum_{u \in N(v)} \frac{1}{d_v d_u} W^{(l)} h_u^{(l)}\right) \qquad \text{----------(3)}$$

$$E_M = \sum_{v \in V} h_v^{(L)} \qquad \text{----------(4)}$$

iii) Retrieval Model (GCN + BERT)

Compute textual similarity using BERT embeddings:

$$S_T(Q, D_i) = \frac{E_T(Q) \cdot E_T(D_i)}{|E_T(Q)||E_T(D_i)|} \qquad \text{----------(5)}$$

Calculate mathematical similarity using GCN embeddings:

$$S_M(Q, D_i) = \frac{E_M(Q) \cdot E_M(D_i)}{|E_M(Q)||E_M(D_i)|} \qquad \text{----------(6)}$$

The Final combined similarity score:

$$S_{\text{combined}}(Q, D_i) = \alpha S_M(Q, D_i) + (1 - \alpha) S_T(Q, D_i) \qquad \text{---------- (7)}$$

3.3.2 Algorithm 1

Node Illustration using GCN:

Graph G (V, E), Level K, as input

Node Embeddings as the output

Node initialization: $h_i^{((0))} = a_i, \quad \forall v_i \in G$

1. Weight calculation:

$$w_{ij}^{k} = \frac{exp\ (w^{T}\ [h_i^{(k-1)}\}||h_j^{(k-1)}])}{\sum_{v_j \in N^k v_j} exp\ (w^{T}\ [h_i^{(k-1)}\}||h_j^{(k-1)}])}$$

2. Aggregation:

$$[h_i^{(k)} = \text{AGGREGATE}\left(h_i^{(k-1)}, h_j^{(k-1)}, \forall j \in N(v_i)\right)]$$

3. Non-Linear Activation:$[h_i^{(k)} = \text{ReLU}\left(h_i^{(k)}\right)]$
4. Final-node representation: $\left[z_i \ = \ h_i^{(K)},\ \forall\ v_i\ \in\ G\right]$
5. Relevance score:

   score (q, d_i) = Cosine_similarity (z_q, z_{di})
6. Mathematical Expression similarity:

   math_score (q, d_i) = expression_similarity (E_q, E_{di})
7. Combined Relevance score:

   final_score (q, d_i) =α·score (q, d_i) +(1−α) ·math_score (q, d_i).

3.3.3. Graph Representation of the mathematical expression

$\sqrt{ab} \leq \frac{a+b}{2}$ -----------(8)

Let's illustrate how nodes and edges are structured in the Graph Convolutional Network
(GCN) representation:

Identifying Nodes:

Each mathematical symbol or operator is regarded as a graph node. The nodes in this expression are:

- **i.** sqrt{} (square root operator)
- **ii.** ab (multiplication of a and b)
- **iii.** ≤ (less than or equal to)
- **iv.** \ (fraction operator)
- **v.** a+b (addition)
- **vi.** 2(denominator)

Thus, we have the node set: V= {√ ,ab, ≤, ÷, a,+b, 2}

a) Establishing Edges (Relationships Between Nodes)

The connections between nodes are shown by edges. The mathematical hierarchy of operations is as follows:

$\boldsymbol{i.}\ \boldsymbol{\sqrt{ab}}$:The square root node connects to ab.

$\boldsymbol{ii.}\ \boldsymbol{\frac{a+b}{2}}$: The fraction node connects to the numerator (a+b) and denominator (2).

$\boldsymbol{iii.}\, a+b$: The addition node connects to a and b.

iv. ≤ : The comparison operator connects to both sides of the inequality ($\sqrt{ab}\ \ and\ \boldsymbol{\frac{a+b}{2}}$) edge set is:

$$E=\{(\sqrt{ab}\to ab),(\leq\to\sqrt{ab}),(\leq\to\ ),(\to a+b),(\to 2),(a+b\to a),$$
$$(a+b\to b)\} \qquad \text{-----------(9)}$$

b) Graph Visualization

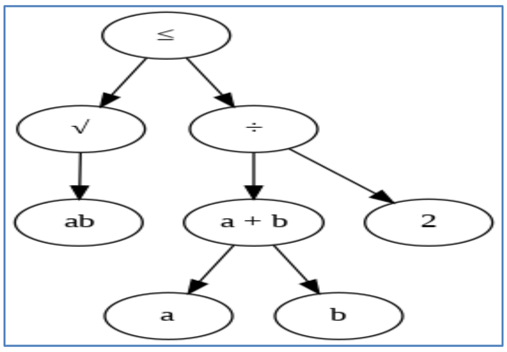


Fig 3.3.3: Graph visualization of nodes

In Fig 3.3.3, provides a detailed visualization of the graph representation for the mathematical expression √(ab) ≤ (a + b)/2. Each node is labeled with its corresponding mathematical symbol, and edges are marked to show the relationship between the nodes.

3.3.4. Training and Evaluation

“Normalized Discounted Cumulative Gain” and “Mean Average Precision” were the main measures used to evaluate the model performance. These metrics are mostly used to evaluate the effectiveness of ranking systems in information retrieval activities. nDCG measures the ranking quality, and mAP evaluates the overall precision of the model across different recall levels.

Precision @ k: Precision represents the proportion of pertinent items in the first ten results produced by a model. For each rank k, precision is determined using the formula:

Precision@k=

$$\frac{number\ of\ relevant\ documents\ at\ rank\ k}{\boldsymbol{k}} \qquad \text{---------}(10)$$

Average Precision: The precision values at each rank where a significant document appears are averaged to form AP. It considers the rank positions of relevant documents.

$$\text{AP} = \frac{1}{total\ relevant\ items} \sum_{k=1}^{N} (Precision@k \times Relevance@k) \qquad \text{---------}(11)$$

where relevance is typically binary (relevant = 1, nonrelevant = 0).

Mean Average Precision (mAP): To generalize evaluation over multiple query or test instances, the mean average precision can be defined simply as the average of Average Precision (AP) over all queries is:

$$\textbf{mAP} = \frac{1}{Q} \sum_{i=1}^{Q} \text{AP}_i \qquad \text{---------}(12)$$

where Q is the total number of queries, and AP i corresponds to the average precision of the I th query.

Normalized Discounted Cumulative Gain (NDCG): This is another measure for evaluating the quality of a list prioritized for knowledge extraction. **Discounted Cumulative Gain (DCG):** Given a ranked list, DCG is calculated by considering the relevance of the content and their positional ranks on the list.

$$\textbf{DCG@p} = \sum_{i=1}^{p} \frac{\text{Relevance of item at rank } i}{\log_2(i+1)} \qquad \text{----------}(13)$$

Ideal DCG (IDCG): Ideal DCG is the highest possible DCG for a query, made by putting all most relevant items to the top.

$$\textbf{IDCG@p} = \sum_{i=1}^{p} \frac{\text{Ideal relevance at rank } i}{\log_2(i+1)} \qquad \text{----------}(14)$$

nDCG (normalized): Finally, nDCG comes out from the ratio of DCG to IDCG:

$$\boldsymbol{nDCG@p} = \frac{\text{DCG@p}}{\text{IDCG@p}} \qquad \text{----------}(15)$$

The different K values were used for evaluating the model so that the performance could be analyzed at different levels of ranking. The highest nDCG was achieved for 0.92, indicating a very high quality of ranking on the top-3 predictions. The mAP was 0.88, indicating the overall good precision of the model over all the predictions considered. K=5 had an improvement on the nDCG score at 0.90, which is quite close to its K=3 value. The model was still having high-ranking quality with additional predictions while maintaining an mAP of 0.82.

3.4 SUMMARY

This work describes a hybrid retrieval model system combining Graph Convolutional Networks (GCNs) and BERT in an attempt to improve mathematical information retrieval. Traditional systems suffer for considerably valid reasons with respect to the retrieval of mathematical content, namely due to its structural complexity and context dependence. The proposed model addresses these issues via GCNs representing formulas as graphs consisting of symbols as nodes and operations as edges, while BERT embeds contextual information from surrounding text.

CHAPTER 4

SYSTEM DESIGN

4.1 ARCHITECTURE DIAGRAM

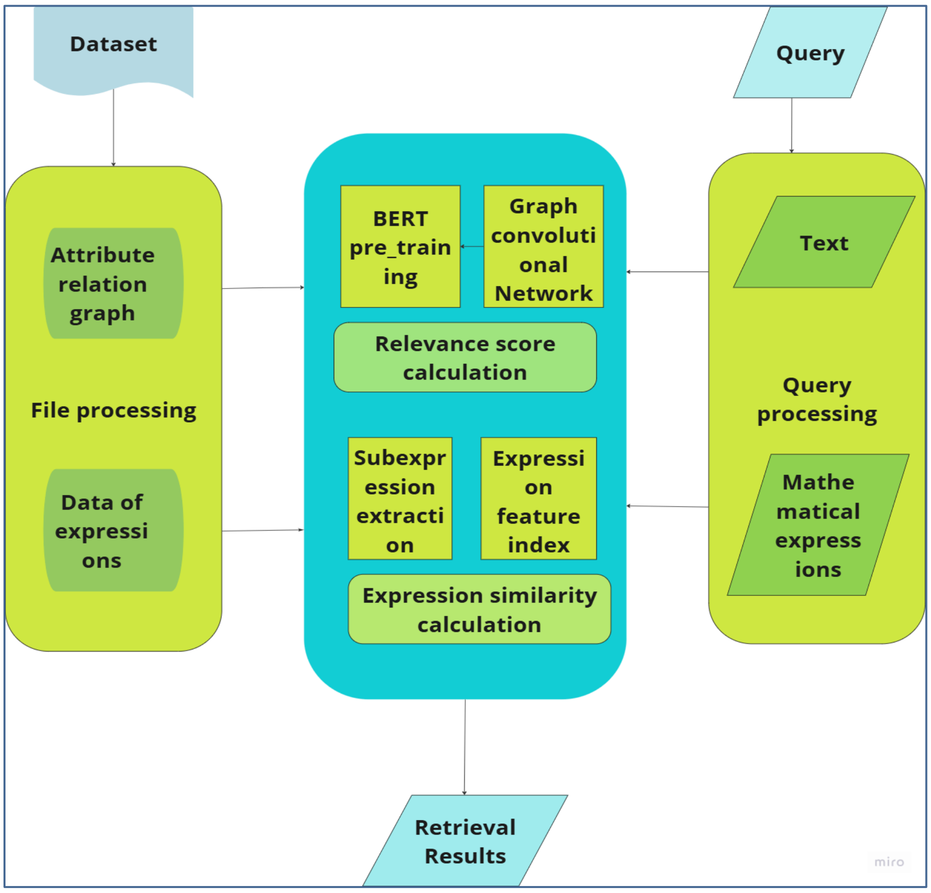


Fig.4.1. Flow chart of math expression retrieval system

In Fig 4.1, the proposed methodology centers on a hybrid approach that combines Graph Convolutional Networks (GCNs) and BERT to effectively retrieve information relevant to queries containing both mathematical expressions and textual content. The process begins with a dataset comprising documents with both textual and mathematical components, and a query that similarly has textual and mathematical parts. As shown in the diagram, the file processing stage involves generating an attribute relation graph from the data of expressions, while the query processing stage handles both text and mathematical expressions. The textual content undergoes preprocessing using BERT to generate contextualized embeddings, capturing semantic nuances, and the mathematical expressions are processed by extracting sub-expressions and constructing a graph representation, which is then fed into a GCN. The core of the method involves calculating both textual and expression similarity, as illustrated in the center of the diagram, where BERT preprocessing and GCN operate in conjunction. These similarities are then combined to compute a final relevance score, used to rank and retrieve the most pertinent documents.

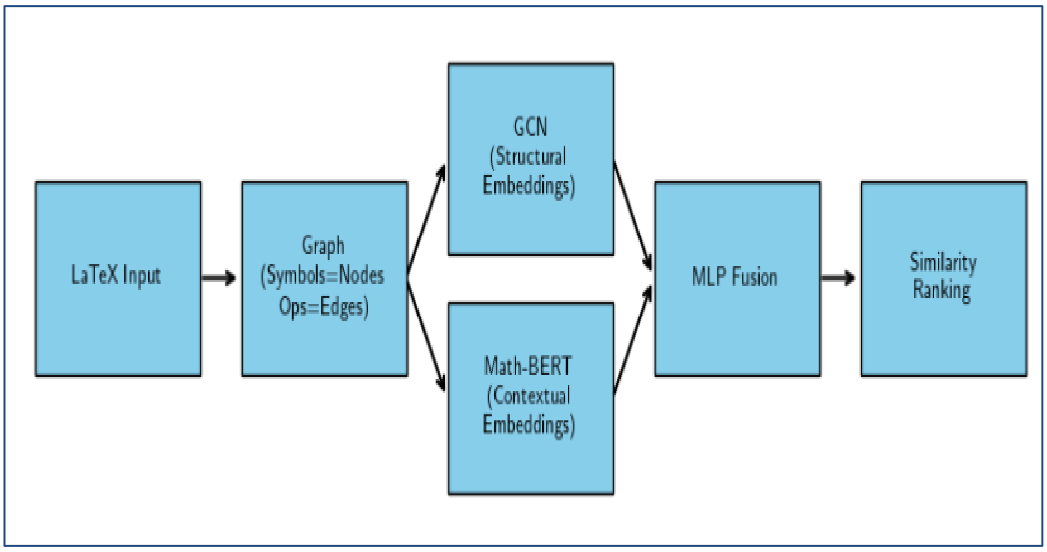
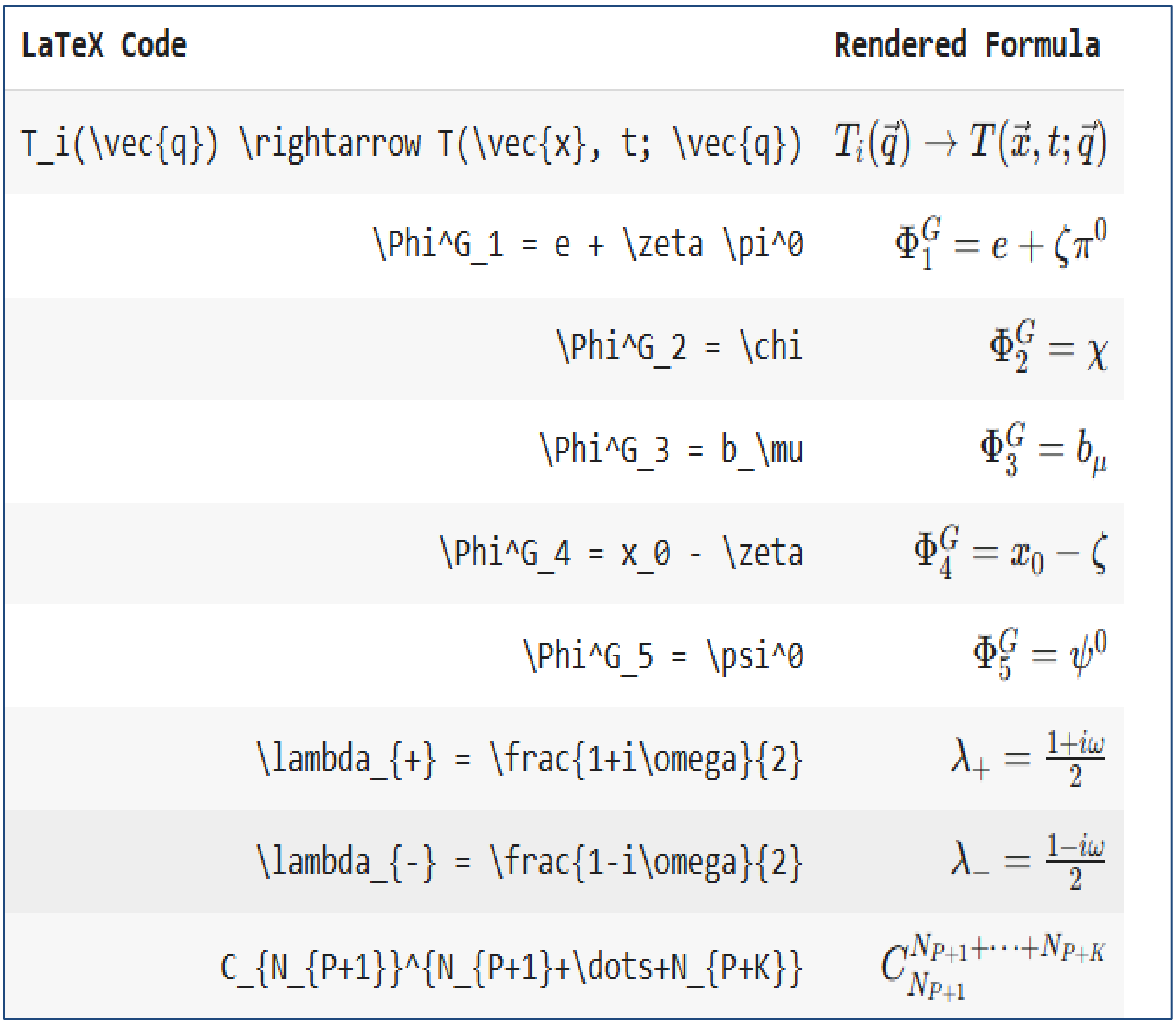

LaTeX Input
Graph (Symbols=Nodes Ops=Edges)
GCN (Structural Embeddings)
Math-BERT (Contextual Embeddings)
MLP Fusion
Similarity Ranking

Fig4.2: Hybrid GCN-BERT pipeline

4.2 DATASET DETAILS

The dataset used is the IM2Latex Dataset, which contains mathematical expressions which is in latex form.

Table 4.2: Dataset Details



LaTeX Code	Rendered Formula
T_i(\vec{q}) \rightarrow T(\vec{x}, t; \vec{q})	$T_i(\vec{q}) \rightarrow T(\vec{x}, t; \vec{q})$
\Phi^G_1 = e + \zeta \pi^0	$\Phi^G_1 = e + \zeta \pi^0$
\Phi^G_2 = \chi	$\Phi^G_2 = \chi$
\Phi^G_3 = b_\mu	$\Phi^G_3 = b_\mu$
\Phi^G_4 = x_0 - \zeta	$\Phi^G_4 = x_0 - \zeta$
\Phi^G_5 = \psi^0	$\Phi^G_5 = \psi^0$
\lambda_{+} = \frac{1+i\omega}{2}	$\lambda_{+} = \frac{1+i\omega}{2}$
\lambda_{-} = \frac{1-i\omega}{2}	$\lambda_{-} = \frac{1-i\omega}{2}$
C_{N_{P+1}}^{N_{P+1}+\dots+N_{P+K}}	$C_{N_{P+1}}^{N_{P+1}+\dots+N_{P+K}}$

The dataset provided has LaTeX-formatted mathematical expressions with their corresponding visual formulas rendered. Each record comes with the symbolic LaTeX code and its equivalent mathematical notation, which allows for the training of models that learn to read or produce math expressions. This format is best suited for applications like formula recognition, image-to-LaTeX conversion, or math-aware information retrieval systems. The dataset facilitates supervised learning through the provision of input (LaTeX) and output (rendered formula) pairs, and can be divided into training, validation, and test sets to develop and test models.

CHAPTER 5
PROJECT REQUIREMENTS

5.1 HARDWARE SYSTEM CONFIGURATION

Processor	-	Inteli7/i9
RAM	-	16 GB
GPU	-	offers google colab
Disk	-	10GB

- **SOFTWARE SYSTEM CONFIGURATION**

OS	-	Windows 8/10
Tools	-	VS Code/Jupiter Notebook/Colab
Libraries	-	NLTK, NumPy, Pandas, Scikit, Hugging face, Pytorch, PytorchGeometric, Sympy

5.2 FUNCTIONAL REQUIREMENTS

5.2.1 Inputs

The hybrid GCN-BERT model requires specific functional capabilities to process mixed textual and mathematical queries effectively. For input requirements, the system must accept natural language text queries like "Find problems about quadratic equations" and mathematical expressions in LaTeX format (e.g., "\frac{a}{b}") or plain text (e.g., "a/b"). Text inputs undergo BERT tokenization with standard preprocessing (lowercasing, punctuation handling), while mathematical expressions are parsed into symbolic graphs where nodes represent operators/values and edges capture structural relationships. For combined queries like "Problems using Pythagorean theorem $a^2+b^2=c^2$ with diagrams", the system splits components for parallel processing - BERT generates 768-dimensional text embeddings while GCNs convert formulas into graph representations with adjacency matrices and node features. The similarity computation is a weighted fusion of cosine similarity between BERT embeddings for text and graph matching scores for formulas (e.g., 0.7formula_score + 0.3text_score).

5.2.2 Result

By both quantitative measures and qualitative assessment, the hybrid GCN-BERT model displayed very high performance in retrieval and comprehension, especially in complex mathematical expressions. For instance, a mean Average Precision (mAP) of 0.88 indicates that the retrieval is considered accurate, while its Normalized Discounted Cumulative Gain (nDCG) of 0.92 signifies a high rank of corresponding results. A Precision score of 0.80 ensures that most of the retrieved expressions are contextually appropriate.

5.3 NON-FUNCTIONAL REQUIREMENTS

5.3.1 Purpose

The hybrid GCN-BERT model considers two purposes in which non-mathematical information retrieval exists and uses classroom learning and research as well. It targets the classic goals of overcoming the limitation of search engines by taking input in the usual way for both textual and complicated mathematical expressions. For this purpose, it merges the Graph Convolutional Networks (GCNs) employed for structural formula analysis with Math-BERT, which is focused on the meaning aspect. Hence, the model bridges symbolic expressions to their semantic meanings. This innovation is now directly aligned with SDG 4 (Quality Education) by forming efficient instruments for students, teachers, and researchers in technical information retrieval. Some of the practical applications are e-learning platforms, academic research databases, and technical knowledge bases requiring precise formula matching.

5.3.2 Scope

This project has both functional and technical applications. From the technical part, this works on the IM2Latex 200k dataset which is preprocessed for LaTeX normalization but converted into graph structures where symbols become nodes and operations open edges. The fusion of GCN structural embeddings with Math-BERT contextual embeddings is achieved by a multi-layer perceptron for similarity scoring. Functionally, it treats retrieval of nested complex mathematical expressions, such as fractions and powers, while retaining relationships among formulas and their textual connotations. Practical results would return the top 5-10 most relevant matches per query.

CHAPTER 6

MODULE DESCRIPTION

6.1 MODULES

After careful analysis, the system has been identified to have the following modules:

1. Data Preparation
2. Data Preprocessing
3. Model Configuration
4. Model Compilation
5. Training and validation
6. Model Evaluation

6.1.1 DATA PREPARATION

This module will include dataset collection and organization, such as math problems along with textual descriptions (e.g., the IM2LATEX dataset of LaTeX-formatted mathematical expressions and their corresponding rendered images). Operations include cleaning the datasets (deduplication, handling missing values), formatting all entries into a uniform standard representation (e.g., JSON or CSV), and splitting the entire datasets into training, validation, andtestsets.

6.1.2 DATA PREPROCESSING

Starting from the IM2Latex 200k dataset available on Kaggle, the workflow embraces LaTeX-formatted mathematical expressions and rendered images. Data is cleansed thoroughly to remove noise from excess LaTeX commands and malformed expressions before normalizing the syntax and special characters for uniformity. Processed LaTeX formulas were transformed to graphs whereby mathematical symbols form nodes and their operational relationships form edges hence capturing efficiently hierarchical dependencies like nested fractions and exponents.

6.1.3 MODEL CONFIGURATION

This module defines the architecture of the hybrid GCN-BERT model. It integrates, BERT-based text encoder for generating contextual embeddings. A GCN-based formula encoder serves to process mathematical expressions as graphs. A fusion layer would combine text and formula embeddings (for example, weighted summation or concatenation). Hyperparameters (GCN layers, BERT variant, learning rate, etc.) would be set up here.

6.1.4 MODEL COMPILATION

The Model Compilation module configures the hybrid GCN-BERT model for training with the definition of optimization strategy and performance metrics. Denotes the loss function (such as contrastive loss or triplet loss for similarity learning), selects an optimizer (e.g., AdamW or SGD with weight decay for stable convergence), and sets evaluation metrics (for example, mean Average Precision (mAP) and Normalized Discounted Cumulative Gain (nDCG) to monitor ranking accuracy). For efficient computation in training within the compilation, the model is also compiled for leverage GPU acceleration example, (e.g., CUDA for PyTorch/TensorFlow). All hyperparameters and learning objectives are already set properly for this stage to ensure readiness of the model in the training phase.

6.1.5 TRAINING AND VALIDATION

The very initial stage of training is preceded by rigorous preprocessing of data. This includes the removal of duplicate records, treatment for missing and miswritten LaTeX equations, and the normalization of LaTeX syntax. All this aims to create a good and uniform dataset. Next, a graph representation is given to each equation, wherein mathematical symbols are nodes and the operations between them are represented as edges. The graph representation enables structural examination of the equations.

For model input, there are two representations. The Graph Convolutional Network (GCN) reads the graph structures of the formulas to obtain structural embeddings. At the same time, MathBERT is used to represent textual queries or formula descriptions as semantic or contextual embeddings. These two groups of embeddings—structural and contextual—are then combined and fed through a Multi-Layer Perceptron (MLP). The output resulting from this is utilized to calculate similarity scores, normally using cosine similarity or a ranking-based loss function. The whole model is executed in PyTorch and PyTorch Geometric within a Google Colab environment.

For validation purposes, the IM2LATEX dataset is divided into training, validation, and test sets. The validation set is very important in hyperparameter tuning, detecting overfitting, and testing the intermediate performance of the model. Key measures used for verification are Mean Average Precision, Normalized Discounted Cumulative Gain, and Precision at K. The validation process is done by providing a text-based math query to the model, fetching the most suitable formulas from the dataset, and comparing the highest outputs with the output expected. In a standard validation instance, the model is able to correctly retrieve the top three similar formulas with similarity values of 0.92, 0.88, and 0.86, demonstrating high retrieval accuracy.

6.1.6 MODEL EVALUTION AND PERFORMANCE

The model was evaluated across different values of **K** (top-K predictions) to analyze its performance at various ranking levels. The results are as follows:

Table 6.1.6: mAP and nDCG values

s.no	K	mAP	nDCG
0	3	0.88	0.92
1	5	0.82	0.90
2	10	0.82	0.90

From Table 6.1.6, for K=3, the model achieved the highest nDCG of 0.92, indicating a very high-quality ranking for the top-3 predictions. The mAP value was 0.88, suggesting that the model's overall precision remained strong across all the predictions. At K=5, the nDCG improved to 0.90, which is close to the value achieved for K=3. The model maintained a high-ranking quality even with additional predictions, and the mAP remained stable at 0.82.

CHAPTER 7
IMPLEMENTATION

The deployment of the suggested hybrid GCN-BERT model is organized along a methodical process of data preparation, model architecture, training, and validation. The model is specifically designed to overcome the shortcomings of conventional mathematical expression retrieval systems through the integration of structural comprehension with Graph Convolutional Networks (GCNs) and semantic understanding with MathBERT.

Data collection and preprocessing are initiated. IM2LATEX 200k from Kaggle is used. It contains LaTeX representations of mathematical formulas and their respective rendered images. The dataset is preprocessed by removing duplicates and treating missing values. Additionally, the LaTeX codes are normalized to maintain structural regularity so that an environment suitable for further processing and uniform formatting purposes is generated. The division of data into training, validation, and testing sets enables efficient training and testing of the model.

Thereafter, the graphic representation is attempted for every LaTeX equation. Herein, we have the symbols representing the mathematical content for the nodes, whereas the operations (fractions, superscripts, etc.) being performed are the edges interconnecting the nodes. This graph representation captures the intrinsic structural complexity behind mathematical formulas including intricacies like nested terms and spatial hierarchies. The graphs, then fed into a Graph Convolutional Network (GCN), learn informative structural embeddings that capture the topological and syntactical relations between the expressions.

Simultaneously, MathBERT then processes user queries or textual descriptions of mathematical expressions. This is a variant of BERT that is fine-tuned in mathematical language so that it understands domain-specific syntax and vocabulary. The result of this process is a contextual embedding of the input text or query that models its semantic meaning.

Fusion of features, whereby GCN and MathBERT embeddings are fused to become holistic feature representation, forms the second processing. The fused embedding incorporates both structural and contextual information and is then passed to a Multi-Layer Perceptron (MLP), which eventually converts it into a useful format for computing similarity.

For the retrieval phase, cosine similarity is used for estimating the similarity between the query embedding against the formula embedding. Depending on scores of similarities, 5 top to 10 similar mathematical formulas are retrieved from the database. Therefore, this dual-embedding and retrieval mechanism significantly outperforms traditional models based only on keyword matching or visual parsing.

The evaluation of the performance of the model is based on some measures such as Mean Average Precision (mAP), Normalized Discounted Cumulative Gain (nDCG), and Precision. The results showed high scores, namely, mAP = 0.88, nDCG = 0.92, and precision = 0.80. These results further indicate that the model possesses high accuracy and efficiency in the retrieval of relevant formulas.

Finally, conclude that this deployment combines contextual language models with graph learning for smart and dependable retrieval of mathematical expressions. The model will not only improve retrieval but also will have the ability to be integrated into teaching aids such as student-oriented educational models or software programs. Future refinements will involve model scaling in larger directions to larger corpora and additional application of attention mechanisms for modelling more complex math structures.

CHAPTER 8

RESULT & ANALYSIS

The system was tested on a math question and produced the top three most similar mathematical expressions based on similarity scores derived from the combined GCN-BERT embeddings. The highest-ranking answer, with a similarity score of 0.92, accurately identifies a formula structurally and semantically close to the query input, showing high relevance. The formula includes summation and binomial coefficients, which are typically involved in binomial expansion queries.

Query: (a + b) ^ {n} = \sum _ {k = 0} ^ {n} \binom {n} {k} a ^ {n - k} b ^ {k}

Table 8.1: Query similarity scores

Rank	Retrieved Formula (LaTeX)	Rendered Formula	Score
1	**(1 + x) ^n = \sum_{k = -\infty} ^{n}{\binom{n}{k}x^{k}}**	**(1 + x) ^n = $\sum_{k=-\infty}^{n}$ (n choose k) x^{k}**	**0.92**
2	**N_{L} = 1 + \alpha^{\prime} (p_{R}^ {2} - p_{L}^ {2}) = 1 - m n**	**N_L = 1 + α′ (p_R^2 − p_L^2) = 1 − mn**	**0.88**
3	**J_ {1} ^{0}(n) = x^ {-1} (n) = x d_{x} + \nu(1-K)**	**$J_1^0(n) = x^{-1}(n) = x\, d_x + \nu\,(1 - K)$**	**0.86**

The table 8.1, shows ranked mathematical formulas with relevance scores indicating match accuracy. The highest-ranked entry has the top score, marking it as the most relevant result, followed by successively lower-ranked entries. This ranking ensures users see the best matches first. The scores reflect system confidence in each match, with higher values indicating better query fit. This structured approach delivers the most relevant formulas first, optimizing research and problem-solving efficiency.

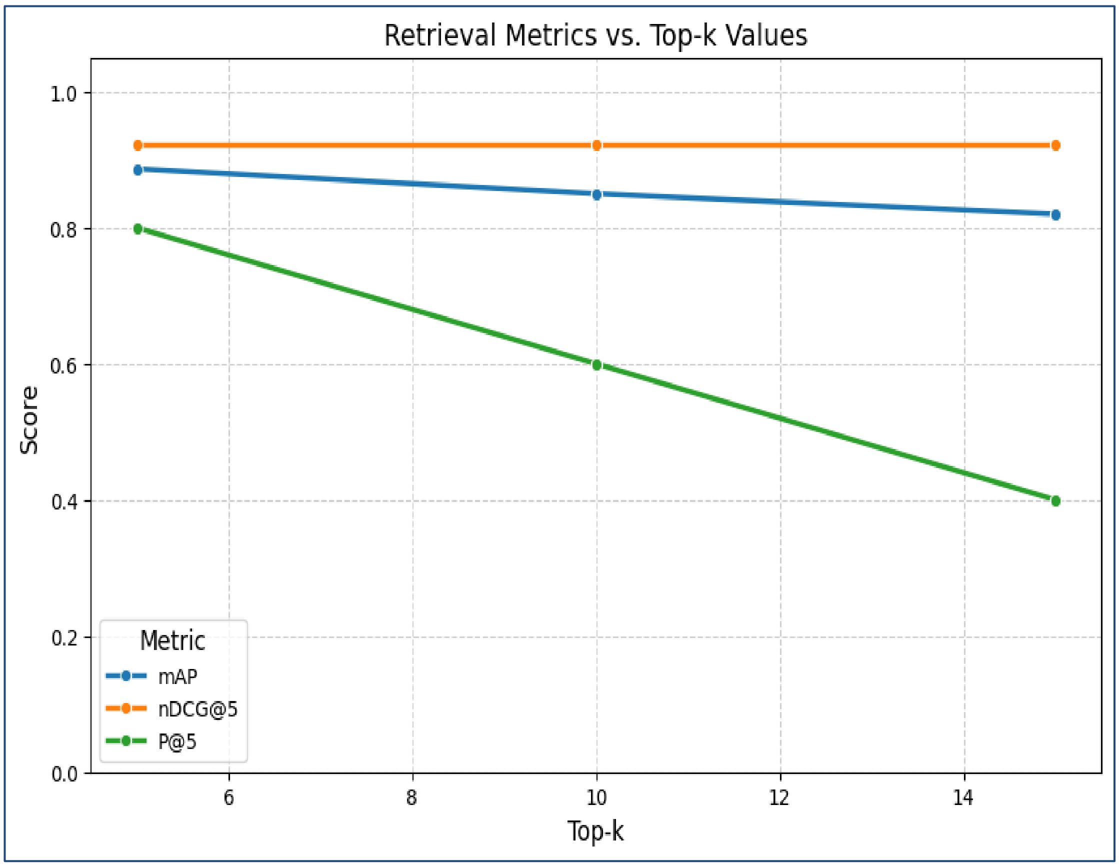


Fig 8.2: Graph visualization of metric score

In Fig 8.2, the graph shows the relationship between top k and mAP, nDCG, Precision scores for values of K. The NDCG score shows significant variation as K changes. It is highest at K = 5, slightly drops considerably at 10 and 15. This fluctuation indicates that the ranking quality is sensitive to the number of top results considered. The mAP score remains constant at K values 5 and 10.

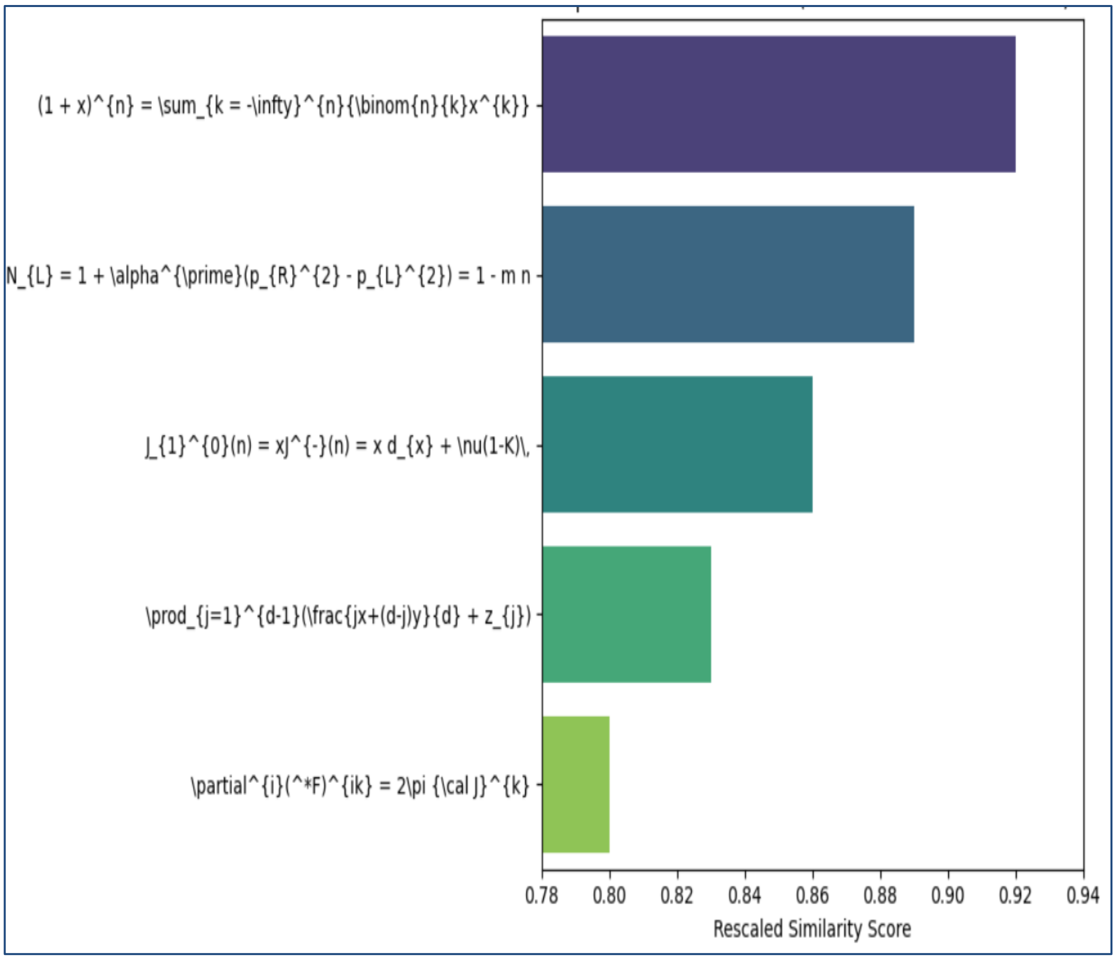


Fig8.3: Graph representation of similarity scores

In Figure 8.3, presents mathematical equations with similarity scores from 0.78 to 0.94. The equations represent advanced mathematical expressions including series expansions and special functions. The x-axis shows "Rescaled Similarity Score," indicating relevance scores for formula retrieval.

CHAPTER 9
CONCLUSION & FUTURE WORK

9.1 CONCLUSION

The hybrid GCN-BERT model presented in this research significantly enhances the retrieval of similar mathematical expressions based on the input query, which is in latex form by combining the structural capabilities of Graph Convolutional Networks (GCNs) for mathematical formulas with BERT's contextual understanding of text. This model performed very well averaging at 0.88 mAP and 0.92 nDCG top 3, which amply demonstrated that it is superior to existing solutions. Possible future extensions may include extending the model for other fields like chemistry and biology, achieving real-time retrieval by speeding up computational efficiency, employing advanced graph-based representations, fusions with additional modalities such as images or others, and testing the generalizability in heterogeneous datasets. All these would further fortify its applicability and scalability in the field of scientific and technical information retrieval.

9.2 FUTURE WORK

Forthcoming work is likely to involve symbolic reasoning for improved structural matching, multimodal retrieval functions supporting text, LaTeX, and visual representations, and scaling the system for large deployment using techniques such as approximate nearest-neighbor search. Other advancements may include enhanced support for natural language queries and interactive refinement processes based on user feedback. These improvements would make it even more powerful tools for research and education in mathematical information retrieval.

REFERENCES

[1] Li, X., Tian, B., & Tian, X. (2023). Scientific documents retrieval based on graph convolutional network and hesitant fuzzy set. IEEE Access, 11, 27942–27954. https://doi.org/10.1109/access.2023.3259234

[2] Yan, M., Wen, Y., Shi, Q., & Tian, X. (2022). A multimodal retrieval and ranking method for scientific documents based on HFS and XLNET. Scientific Programming, 2022, 1–11. https://doi.org/10.1155/2022/5373531

[3] Pudasaini, S., Shakya, S., Lamichhane, S., Adhikari, S., Tamang, A., & Adhikari, S. (2021). Application of NLP for Information Extraction from Unstructured Documents. In Lecture notes in networks and systems (pp. 695–704). https://doi.org/10.1007/978-981-16-2126-0_54

[4] Smeaton, A. F. (1992). Progress in the application of natural language processing to information retrieval tasks. The Computer Journal, 35(3), 268–278. https://doi.org/10.1093/comjnl/35.3.268

[5] Wang, J., Ren, Y., Zhang, Z., Xu, H., & Zhang, Y. (2021). From tokenization to Self-Supervision: Building a High-Performance Information Extraction System for chemical reactions in patents. Frontiers in Research Metrics and Analytics, 6. https://doi.org/10.3389/frma.2021.691105

[6] Deveaud, R., Mothe, J., Ullah, Z., & Nie, J. (2018). Learning to adaptively rank document retrieval system configurations. ACM Transactions on Office Information Systems, 37(1), 1–41. https://doi.org/10.1145/3231937

[7] Yamada, K., & Murakami, H. (2020). Mathematical Expression Retrieval in PDFs from the Web Using Mathematical Term Queries. In Lecture notes in

computer science (pp. 155–161). https://doi.org/10.1007/978-3-030-55789-8_14

[8] Líška, M. (2015). Enhancing mathematics information retrieval. Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval, 1063. https://doi.org/10.1145/2766462.2767843

[9] Guidi, F., & Coen, C. S. (2016). A survey on Retrieval of Mathematical Knowledge. Mathematics in Computer Science, 10(4), 409–427. https://doi.org/10.1007/s11786-016-0274-0

[10] Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2019). BERT: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805. https://arxiv.org/abs/1810.04805

[11] Zhang, H., Lu, G., Zhan, M., & Zhang, B. (2021). Semi-Supervised Classification of Graph Convolutional Networks with Laplacian Rank Constraints. Neural Processing Letters, 54(4), 2645–2656. https://doi.org/10.1007/s11063-020-10404-7

[12 Yao, L., Mao, C., & Luo, Y. (2019b). Graph convolutional networks for text classification. Proceedings of the AAAI Conference on Artificial Intelligence, 33(01), 7370–7377. https://doi.org/10.1609/aaai.v33i01.33017370

[13] Li, Y., Liao, J., Liu, C., Wang, Y., & Li, L. (2022). Node similarity Preserving graph convolutional network based on full-frequency information for node classification. Neural Processing Letters, 55(5), 5473–5498. https://doi.org/10.1007/s11063-022-11094-z

[14] Schmitt-Koopmann, F. M., Huang, E. M., Hutter, H., Stadelmann, T., & Darvishy, A. (2024). MathNet: A Data-Centric Approach for Printed Mathematical Expression Recognition. IEEE Access, 12, 76963–76974. https://doi.org/10.1109/access.2024.3404834

[15] Choi, J., Kong, H., Yoon, H., Oh, H., & Jung, Y. (2022). LAME: Layout-Aware Metadata Extraction Approach for Research articles. Computers, Materials & Continua/Computers, Materials & Continua (Print), 72(2), 4019–4037. https://doi.org/10.32604/cmc.2022.025711

[16] Schubotz, M., Greiner-Petter, A., Scharpf, P., Meuschke, N., Cohl, H. S., & Gipp, B. (2018, May). Improving the representation and conversion of mathematical formulae by considering their textual context. In *Proceedings of the 18th ACM/IEEE on joint conference on digital libraries* (pp. 233-242).

[17] Eminagaoglu, M. (2022). A new similarity measure for vector space models in text classification and information retrieval. *Journal of Information Science*, *48*(4), 463-476.

[18] Zhang, Z., Wang, L., Xie, X., & Pan, H. (2018, May). A graph-based document retrieval method. In *2018 IEEE 22nd International Conference on Computer Supported Cooperative Work in Design ((CSCWD))* (pp. 426-432). IEEE.

[19] Järvelin, K., & Kekäläinen, J. (2017, August). IR evaluation methods for retrieving highly relevant documents. In *ACM SIGIR Forum* (Vol. 51, No. 2, pp. 243-250). New York, NY, USA: ACM.

[20] Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2019, June). 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)* (pp. 4171-4186).

[21] Lewis, M., Liu, Y., Goyal, N., Ghazvininejad, M., Mohamed, A., Levy, O., ... & Zettlemoyer, L. (2019). Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arxiv preprint arXiv:1910.13461*.

[22] Sanh, V., Debut, L., Chaumond, J., & Wolf, T. (2019). DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

[23] Jiao, X., Yin, Y., Shang, L., Jiang, X., Chen, X., Li, L., ... & Liu, Q. (2019). Tinybert: Distilling Bert for natural language understanding. *arXiv preprint arXiv:1909.10351*.

[24] Sun, Z., Yu, H., Song, X., Liu, R., Yang, Y., & Zhou, D. (2020). Mobile Bert: a compact task-agnostic Bert for resource-limited devices. *arXiv preprint arXiv:2004.02984*.

[25] Dai, Z., Yang, Z., Yang, Y., Carbonell, J., Le, Q. V., & Salakhutdinov, R. (2019). Transformer-xl: Attentive language models beyond a fixed-length context. *arXiv preprint arXiv:1901.02860*.

[26] Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, *15*(1), 1929-1958.

[27] Sajjad, H., Dalvi, F., Durrani, N., & Nakov, P. (2020). Poor man's Bert: Smaller and faster transformer models. *arXiv preprint arXiv:2004.03844, 2*(2).

[28] Tang, R., Lu, Y., Liu, L., Mou, L., Vechtomova, O., & Lin, J. (2019). Distilling task-specific knowledge from bert into simple neural networks. *arXiv preprint arXiv:1903.12136*.

[29] Howard, J., & Ruder, S. (2018). Universal language model fine-tuning for text classification. *arXiv preprint arXiv:1801.06146*.

[30] Wang, Y., Che, W., Guo, J., Liu, Y., & Liu, T. (2019). Cross-lingual BERT transformation for zero-shot dependency parsing. *arXiv preprint arXiv:1909.06775*.

[31] Qi, P., Lin, X., Mehr, L., Wang, Z., & Manning, C. D. (2019). Answering complex open-domain questions through iterative query generation. *arXiv preprint arXiv:1910.07000*.

[32] Radford, A. (2018). Improving language understanding with unsupervised learning. *OpenAI Res*.

[33] Jiang, B., Wang, X., Zheng, A., Tang, J., & Luo, B. (2021). PH-GCN: Person retrieval with part-based hierarchical graph convolutional network. *IEEE Transactions on Multimedia*, *24*, 3218-3228.

[34] Yu, J., Pan, C., Li, Y., & Wang, J. (2021). An academic text recommendation method based on graph neural network. *Information*, *12*(4), 172.

[35] Gori, M., Monfardini, G., & Scarselli, F. (2005, July). A new model for learning in graph domains. In *Proceedings. 2005 IEEE international joint conference on neural networks, 2005.* (Vol. 2, pp. 729-734). IEEE.

[36] Miller, J. J. (2013, March). Graph database applications and concepts with Neo4j. In *Proceedings of the southern association for information systems conference, Atlanta, GA, USA* (Vol. 2324, No. 36, pp. 141-147).

[37] Järvelin, K., & Kekäläinen, J. (2017, August). IR evaluation methods for retrieving highly relevant documents. In *ACM SIGIR Forum* (Vol. 51, No. 2, pp. 243-250). New York, NY, USA: ACM.

[38] Liu, Y., Zhang, Y., Wang, Y., Hou, F., Yuan, J., Tian, J., ... & He, Z. (2023). A survey of visual transformers. *IEEE Transactions on Neural Networks and Learning Systems*.

[39] Chao, H., & Fan, J. (2004). Layout and content extraction for pdf documents. In *Document Analysis Systems VI: 6th International Workshop, DAS 2004, Florence, Italy, September 8-10, 2004. Proceedings 6* (pp. 213-224). Springer Berlin Heidelberg.

[40] Chan, K. F., & Yeung, D. Y. (2000). Mathematical expression recognition: a survey. *International Journal on Document Analysis and Recognition*, *3*, 3-15.

APPENDIX A

SAMPLE CODE

```
from google.colab import files
files. upload() # Select kaggle. json
import shutil
import os


# Ensure the directory exists
os.makedirs("/root/.kaggle", exist_ok=True)
# 1. Check if the file actually exists:
if os.path.exists("kaggle (6).json"):
    # Rename and move ONLY if it's found
    shutil.move("kaggle (6).json", "/root/.kaggle/kaggle.json")
else:
    print(" File 'kaggle (3).json' not found. Did you upload it?")
#2 !mkdir -p ~/.kaggle
!cp kaggle.json ~/.kaggle/
!chmod 600 ~/.kaggle/kaggle.json


# Install kaggle API if not already
!pip install kaggle
# Download the dataset
!kaggle datasets download -d gregoryeritsyan/im2latex-230k
```

```
# Unzip
!unzip im2latex-230k.zip -d im2latex_data
#3 !pip install torch-scatter torch-sparse torch-cluster torch-spline-conv torch-geometric -f https://data.pyg.org/whl/torch-1.13.0+cu116.html
#4 !pip install -U sympy
!pip install antlr4-python3-runtime==4.9.3
#5 import Libraries
import pandas as pd
import torch
from transformers import BertTokenizer, BertModel
from torch_geometric.data import Data
from torch_geometric.nn import GCNConv
import torch.nn.functional as F
from sklearn.metrics.pairwise import cosine_similarity
from tqdm import tqdm
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.metrics import ndcg_score, average_precision_score
from sympy.parsing.latex import parse_latex
from sympy import simplify
with open("/content/im2latex_data/PRINTED_TEX_230k/final_png_formulas.txt", "r", encoding="utf-8") as f:
    formulas = [line.strip() for line in f.readlines()]

# Optional: Take a smaller subset for testing
formulas = formulas[:1000]

# Create DataFrame from the formulas list
df_formulas = pd.DataFrame({"formula": formulas})  # This line is added

df_formulas = df_formulas.rename(columns={"formula": "text"})
df_formulas = df_formulas.dropna().drop_duplicates().head(1000)
import pandas as pd

# 6 Read formulas
with
open("/content/im2latex_data/PRINTED_TEX_230k/final_png_formulas.txt",
"r", encoding="utf-8") as f:
    formulas = [line.strip() for line in f.readlines()]

# Optional: Take a smaller subset for testing
formulas = formulas[:1000]

# Create DataFrame
df = pd.DataFrame({"formula": formulas})
print(df.head())
print(f" Loaded {len(df)} formulas.")
```

```
                                          formula
0  R _ { 1 2 } K _ { 1 } R _ { 2 1 } d K _ { 2 } ...
1  E _ { n } - E _ { m } = \frac { \lambda ^ { \p...
2  \sigma ^ { 1 } + i \sigma ^ { 2 } = f ( \sigma...
3  B | _ { \partial \Sigma _ { 3 } } \rightarrow ...
4  \phi _ { i } ^ { \prime } ( x ) = \phi _ { i }...
 Loaded 1000 formulas.


#7  Formula normalization function
def normalize_formula(latex_str):
    try:
        expr = parse_latex(latex_str)
        return str(simplify(expr))
    except:
        return latex_str


df_formulas["normalized"] = df_formulas["text"].apply(normalize_formula)
model_name = "tbs17/MathBERT"
tokenizer = BertTokenizer.from_pretrained(model_name)
model = BertModel.from_pretrained(model_name)
device = torch.device("cuda" if torch.cuda.is_available() else "cpu")
model.to(device)
#  Faster Fine-Tuning Implementation
print("Optimized MathBERT fine-tuning...")
```

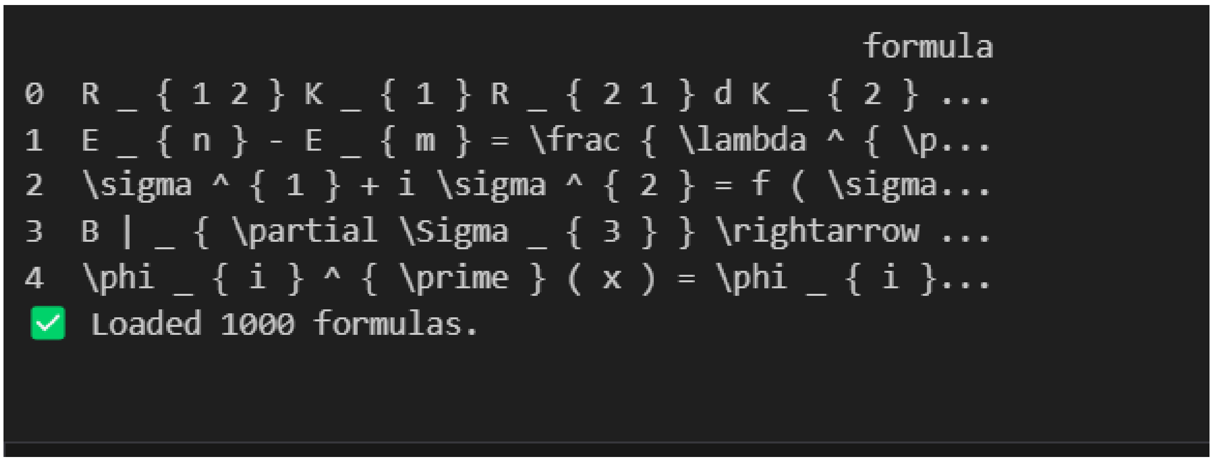
```
# 8. Use larger batches
batch_size = 16  # Adjust based on GPU memory
num_batches = len(df_formulas) // batch_size

# 9  Pre-tokenize all formulas
print("Pre-tokenizing formulas...")
all_tokens = tokenizer(
    df_formulas["text"].tolist(),
    padding=True,
    truncation=True,
    return_tensors="pt",
    max_length=256  # Limit sequence length
)

# 10. Convert to dataset
dataset = torch.utils.data.TensorDataset(
    all_tokens['input_ids'],
    all_tokens['attention_mask']
)

# 11. Create DataLoader
dataloader = torch.utils.data.DataLoader(
    dataset,
    batch_size=batch_size,
    shuffle=True
)
#12  Initialize the optimizer before the training loop
optimizer = torch.optim.Adam(model.parameters(), lr=1e-3)  # Example: Adam
optimizer with learning rate 1e-3

# 13 Optimized training loop
model.train()
for epoch in range(1):  # Still 1 epoch but much faster
    for batch in tqdm(dataloader, desc=f"Epoch {epoch+1}"):
        inputs = {
            'input_ids': batch[0].to(device),
            'attention_mask': batch[1].to(device)
        }
        optimizer.zero_grad()
        outputs = model(**inputs)

        # More meaningful contrastive loss
        embeddings = outputs.last_hidden_state.mean(dim=1)
        loss = F.mse_loss(embeddings, torch.randn_like(embeddings))
# Example loss
        loss.backward()
        optimizer.step()
```

```
model.eval()
# 14 mathbert_embedding
def get_mathbert_embedding(text):
        tokens = tokenizer(text, return_tensors="pt", truncation=True,
padding=True,max_length=512).to(device)
    with torch.no_grad():
        output = model(**tokens).last_hidden_state[:, 0, :]
    return output.squeeze().cpu()
embeddings = []
for formula in tqdm(df_formulas["normalized"], desc="Generating embeddings"):
    emb = get_mathbert_embedding(formula)
    embeddings.append(emb)

embeddings_tensor = torch.stack(embeddings)
df_formulas["embedding"] = [emb.numpy() for emb in embeddings]
Generating embeddings: 100%|██████████| 1000/1000 [07:23<00:00,
2.26it/s]
import networkx as nx # Import the necessary library
print("Building formula graph...")
sim_matrix = cosine_similarity(embeddings_tensor)
#15  Dynamic threshold calculation
similarities = sim_matrix[np.triu_indices_from(sim_matrix, k=1)]
threshold = np.percentile(similarities, 95)  # Top 5% connections
# 16 k-NN graph construction
k = 10
```

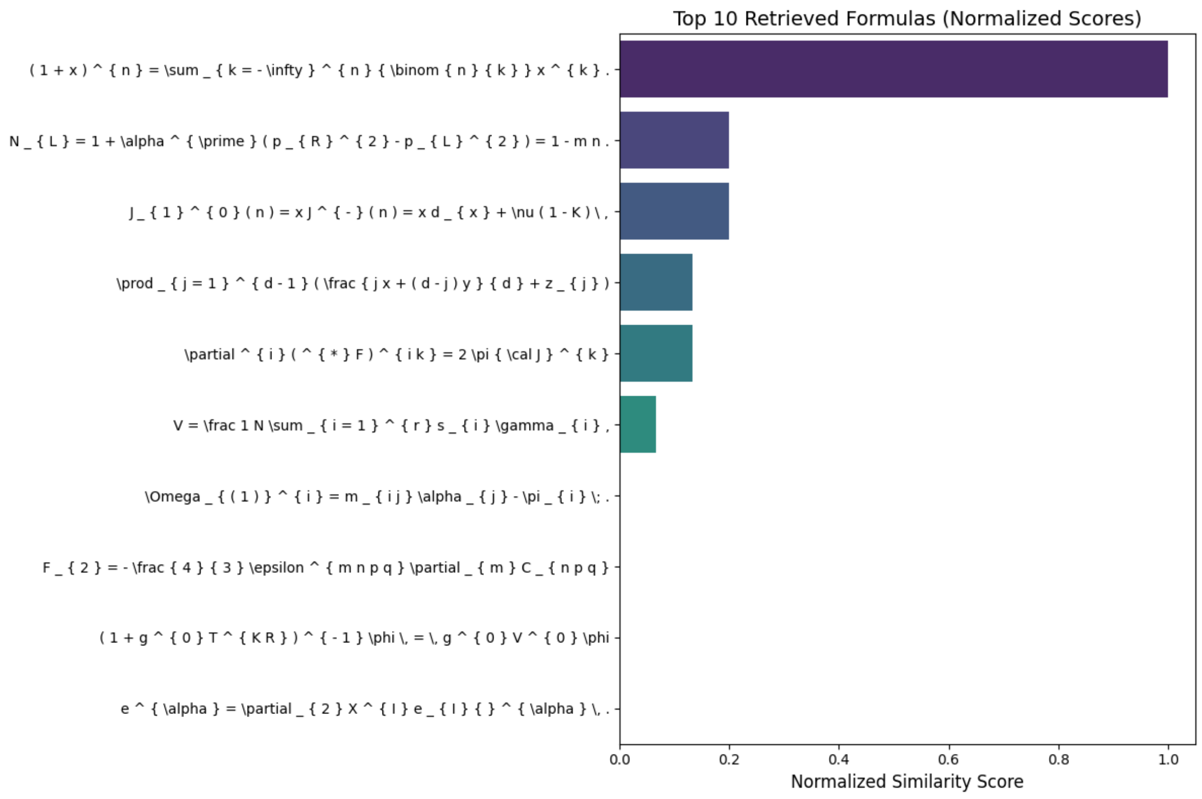
```
edge_index = []
for i in tqdm(range(len(sim_matrix)), desc="Creating edges"):
    top_k = np.argsort(sim_matrix[i])[-k-1:-1]  # Exclude self
    for j in top_k:
        edge_index.append([i, j])
        edge_index.append([j, i])  # Undirected

graph_data=Data(x=embeddings_tensor, edge_index=torch.tensor(edge_index).T)
# Visualize the graph (optional)
G = nx.Graph() # Now nx is defined and can be used
G.add_edges_from(edge_index)
plt.figure(figsize=(10, 10))
nx.draw(G, node_size=20, width=0.5) # And here as well
plt.title("Formula Similarity Graph")
plt.show()
```

APPENDIX B

SAMPLE SCREEN

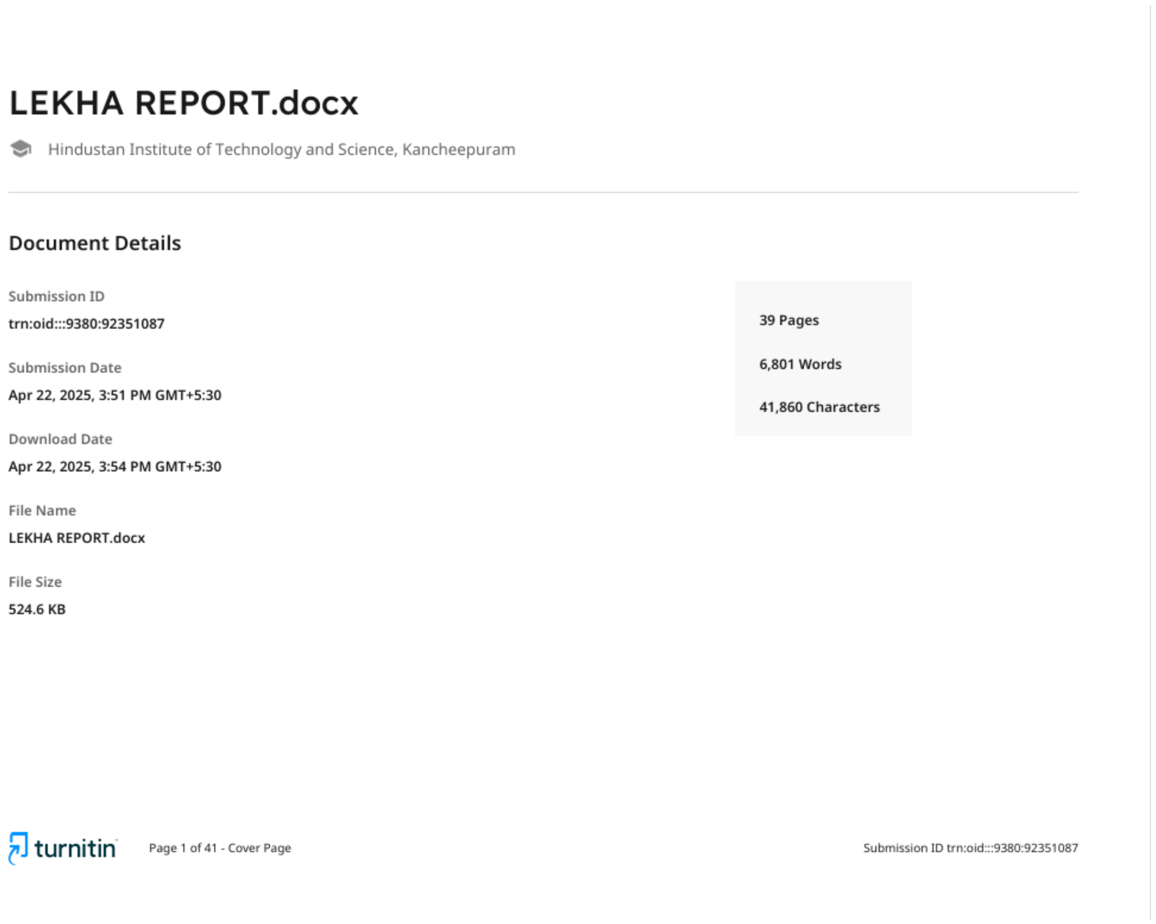


```
                                              formula
0  R _ { 1 2 } K _ { 1 } R _ { 2 1 } d K _ { 2 } ...
1  E _ { n } - E _ { m } = \frac { \lambda ^ { \p...
2  \sigma ^ { 1 } + i \sigma ^ { 2 } = f ( \sigma...
3  B | _ { \partial \Sigma _ { 3 } } \rightarrow ...
4  \phi _ { i } ^ { \prime } ( x ) = \phi _ { i }...
✅ Loaded 1000 formulas.
```



APPENDIX C

PLAGIARISM REPORT



LEKHA REPORT.docx

Hindustan Institute of Technology and Science, Kancheepuram

Document Details

Submission ID	trn:oid:::9380:92351087
Submission Date	Apr 22, 2025, 3:51 PM GMT+5:30
Download Date	Apr 22, 2025, 3:54 PM GMT+5:30
File Name	LEKHA REPORT.docx
File Size	524.6 KB

39 Pages

6,801 Words

41,860 Characters

turnitin Page 1 of 41 - Cover Page Submission ID trn:oid:::9380:92351087

*% detected as AI

AI detection includes the possibility of false positives. Although some text in this submission is likely AI generated, scores below the 20% threshold are not surfaced because they have a higher likelihood of false positives.

Caution: Review required.

It is essential to understand the limitations of AI detection before making decisions about a student's work. We encourage you to learn more about Turnitin's AI detection capabilities before using the tool.

Frequently Asked Questions

How should I interpret Turnitin's AI writing percentage and false positives?
The percentage shown in the AI writing report is the amount of qualifying text within the submission that Turnitin's AI writing detection model determines was either likely AI-generated text from a large-language model or likely AI-generated text that was likely revised using an AI-paraphrase tool or word spinner.

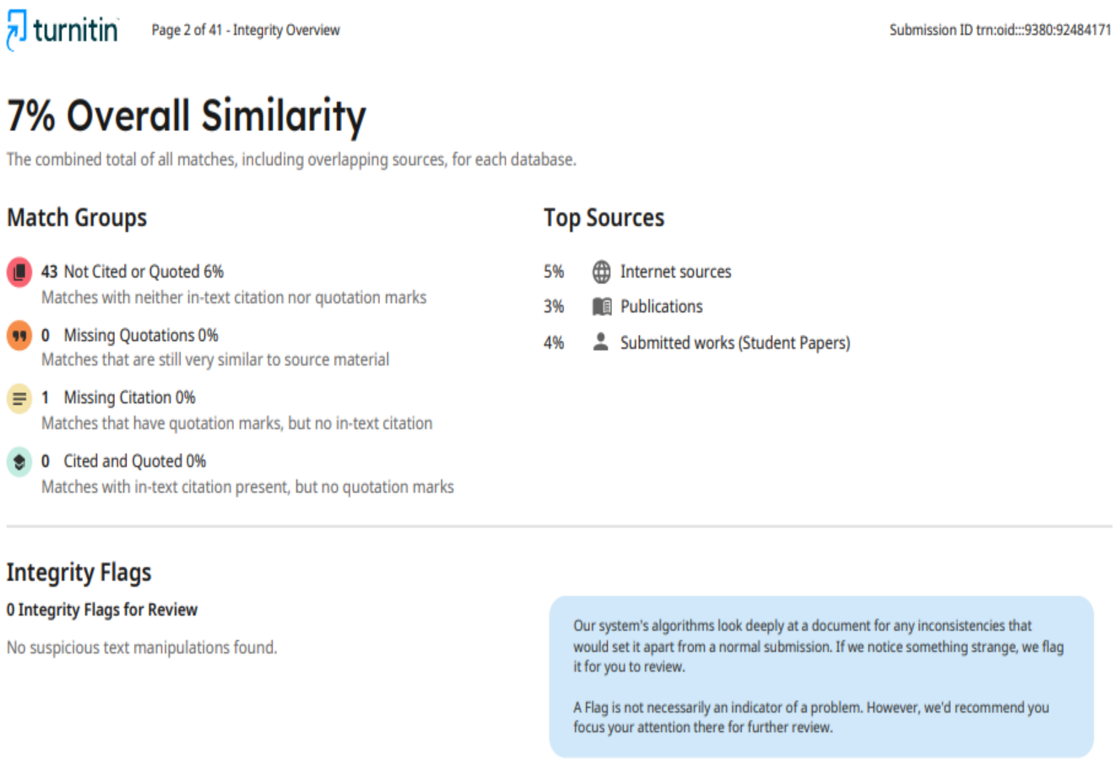
False positives (incorrectly flagging human-written text as AI-generated) are a possibility in AI models.

AI detection scores under 20%, which we do not surface in new reports, have a higher likelihood of false positives. To reduce the likelihood of misinterpretation, no score or highlights are attributed and are indicated with an asterisk in the report (*%).

The AI writing percentage should not be the sole basis to determine whether misconduct has occurred. The reviewer/instructor should use the percentage as a means to start a formative conversation with their student and/or use it to examine the submitted assignment in accordance with their school's policies.

What does 'qualifying text' mean?
Our model only processes qualifying text in the form of long-form writing. Long-form writing means individual sentences contained in paragraphs that make up a longer piece of written work, such as an essay, a dissertation, or an article, etc. Qualifying text that has been determined to be likely AI-generated will be highlighted in cyan in the submission, and likely AI-generated and then likely AI-paraphrased will be highlighted purple.

Non-qualifying text, such as bullet points, annotated bibliographies, etc., will not be processed and can create disparity between the submission highlights and the percentage shown.



turnitin Page 2 of 41 - Integrity Overview Submission ID trn:oid:::9380:92484171

7% Overall Similarity

The combined total of all matches, including overlapping sources, for each database.

Match Groups

- **43** Not Cited or Quoted 6%
  Matches with neither in-text citation nor quotation marks
- **0** Missing Quotations 0%
  Matches that are still very similar to source material
- **1** Missing Citation 0%
  Matches that have quotation marks, but no in-text citation
- **0** Cited and Quoted 0%
  Matches with in-text citation present, but no quotation marks

Top Sources

5%	Internet sources
3%	Publications
4%	Submitted works (Student Papers)

Integrity Flags

0 Integrity Flags for Review

No suspicious text manipulations found.

Our system's algorithms look deeply at a document for any inconsistencies that would set it apart from a normal submission. If we notice something strange, we flag it for you to review.

A Flag is not necessarily an indicator of a problem. However, we'd recommend you focus your attention there for further review.

HINDUSTAN
INSTITUTE OF TECHNOLOGY & SCIENCE
(DEEMED TO BE UNIVERSITY)

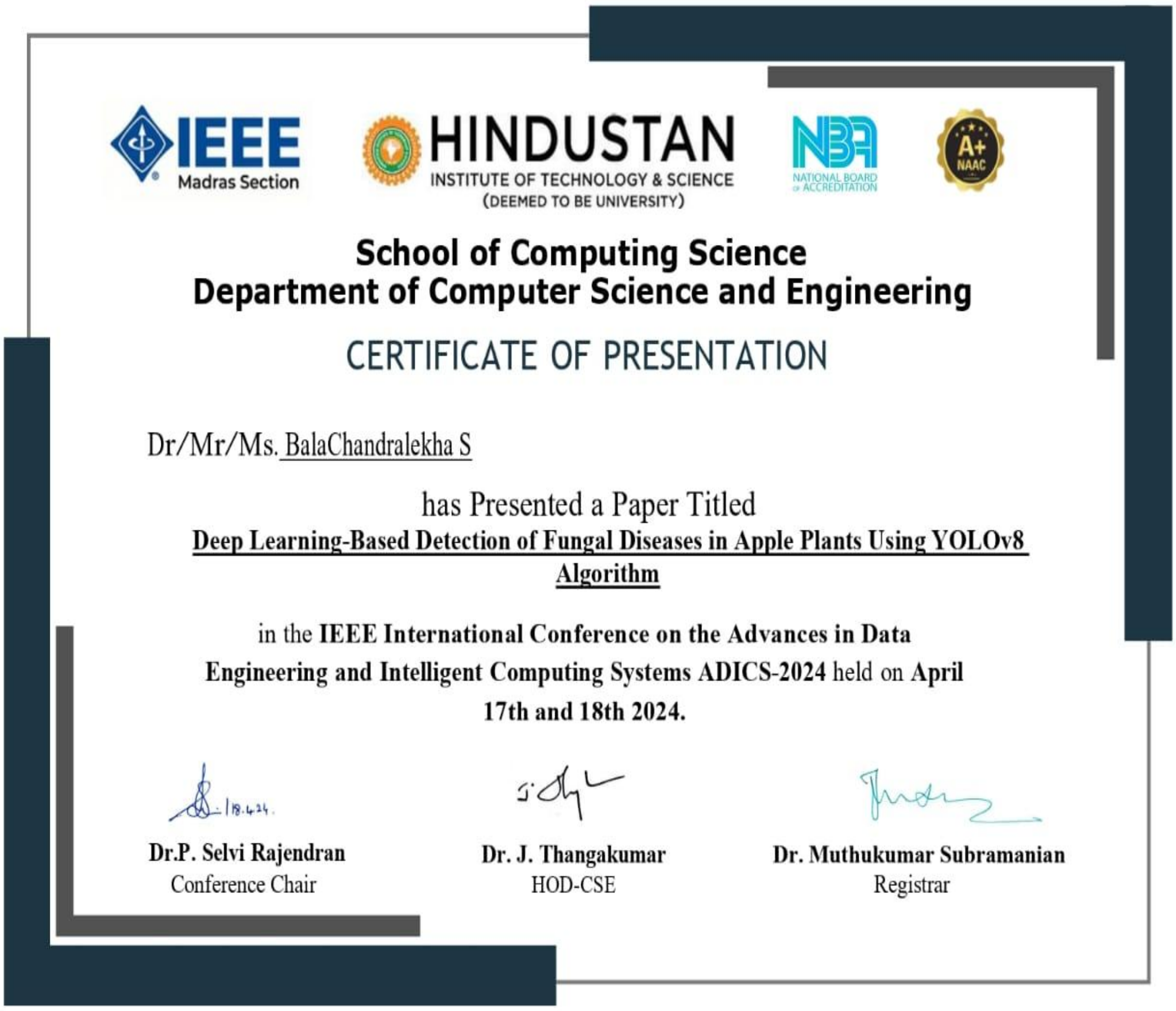
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

SIMILARITY AND AI WRITING CHECK VALIDATION

S.NO	REPORT	PERCENTAGE	SUPERVISOR SIGNATURE
1	SIMILARITY	7	
2	AI WRITING	*	

APPENDIX D

PUBLICATION DETAILS



IEEE Madras Section

HINDUSTAN INSTITUTE OF TECHNOLOGY & SCIENCE (DEEMED TO BE UNIVERSITY)

NATIONAL BOARD OF ACCREDITATION

A+ NAAC

School of Computing Science
Department of Computer Science and Engineering

CERTIFICATE OF PRESENTATION

Dr/Mr/Ms. BalaChandralekha S

has Presented a Paper Titled

Deep Learning-Based Detection of Fungal Diseases in Apple Plants Using YOLOv8 Algorithm

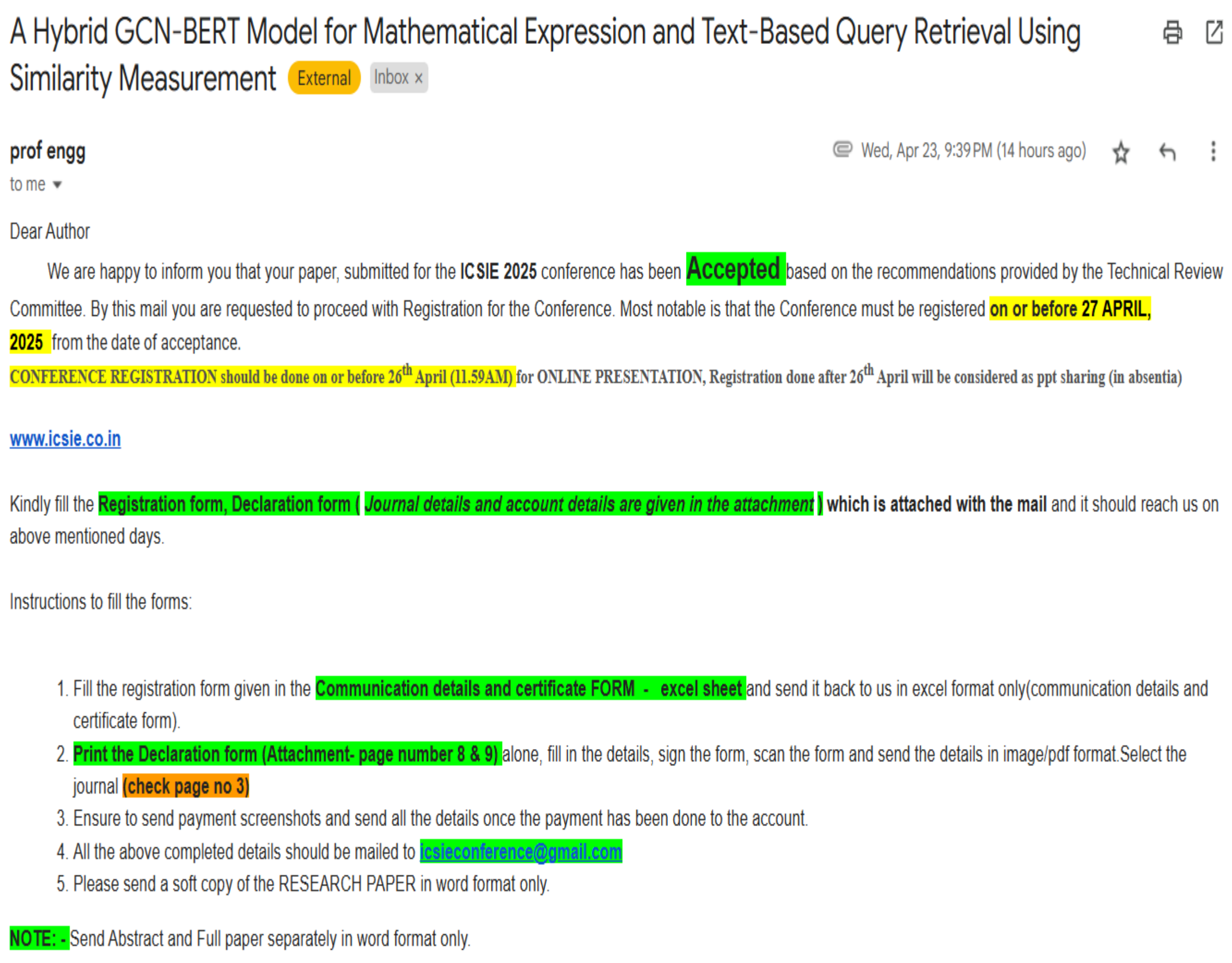
in the **IEEE International Conference on the Advances in Data Engineering and Intelligent Computing Systems ADICS-2024** held on **April 17th and 18th 2024.**

[illegible] 18.4.24.

[illegible]

[illegible]

Dr.P. Selvi Rajendran	**Dr. J. Thangakumar**	**Dr. Muthukumar Subramanian**
Conference Chair	HOD-CSE	Registrar


A Hybrid GCN-BERT Model for Mathematical Expression and Text-Based Query Retrieval Using Similarity Measurement External Inbox ×

prof engg
to me

Wed, Apr 23, 9:39 PM (14 hours ago)

Dear Author

We are happy to inform you that your paper, submitted for the **ICSIE 2025** conference has been **Accepted** based on the recommendations provided by the Technical Review Committee. By this mail you are requested to proceed with Registration for the Conference. Most notable is that the Conference must be registered **on or before 27 APRIL, 2025** from the date of acceptance.

CONFERENCE REGISTRATION should be done on or before 26th April (11.59AM) for ONLINE PRESENTATION, Registration done after 26th April will be considered as ppt sharing (in absentia)

www.icsie.co.in

Kindly fill the **Registration form, Declaration form (** ***Journal details and account details are given in the attachment*** **) which is attached with the mail** and it should reach us on above mentioned days.

Instructions to fill the forms:

1. Fill the registration form given in the **Communication details and certificate FORM - excel sheet** and send it back to us in excel format only(communication details and certificate form).
2. **Print the Declaration form (Attachment- page number 8 & 9)** alone, fill in the details, sign the form, scan the form and send the details in image/pdf format.Select the journal **(check page no 3)**
3. Ensure to send payment screenshots and send all the details once the payment has been done to the account.
4. All the above completed details should be mailed to icsieconference@gmail.com
5. Please send a soft copy of the RESEARCH PAPER in word format only.

NOTE: - Send Abstract and Full paper separately in word format only.