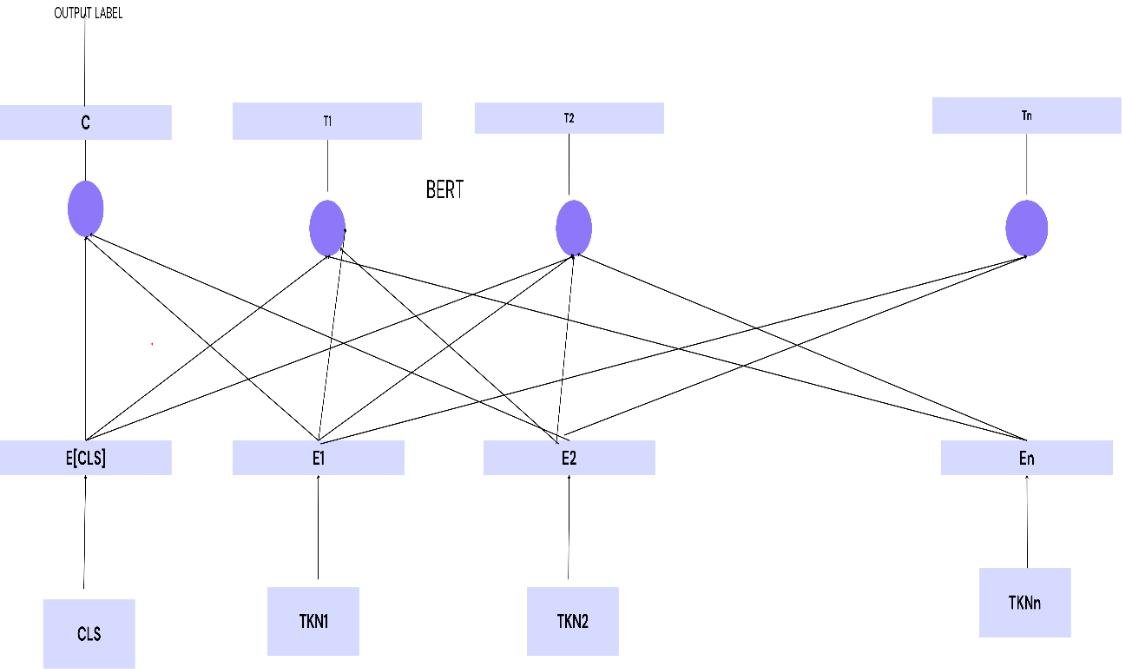
The project describes a hybrid model that combines Graph Convolutional Networks (GCNs) with BERT (Bidirectional Encoder Representations from Transformers) to attempt to radically transform mathematical information retrieval. Traditional search systems are troubled with mathematical content owing to its 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.

### 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.



BERT Model

In the Figure , 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.

1. **Text Preprocessing for BERT**

ET= BERT(Ti)

----------(1)

Where Q, K, and V represent the query, key, and value matrices, respectively,

The SoftMax function normalizes the attention scores.

#### Graph Representation of Mathematical Expressions

#### ----------(2)

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

hv(0)​=one-hot(v)

----------(3)

----------(4)

#### Retrieval Model (GCN + BERT)

Compute textual similarity using BERT embeddings:

----------(5)

Calculate mathematical similarity using GCN embeddings:

----------(6)

The Final combined similarity score:

---------- (7)

**Algorithm 1**

Node Illustration using GCN:

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

Node Embeddings as the output

Node initialization:

1. Weight calculation:
2. Aggregation:
3. Non-Linear Activation:
4. Final-node representation:
5. Relevance score:

score (*q*, *di*​) = Cosine\_similarity (*zq*​, *zdi*​)

1. Mathematical Expression similarity:

math\_score (*q*, *di*​) = expression\_similarity (*Eq*​, *E di*​)

1. Combined Relevance score:

final\_score (*q*, *di*​) =*α*⋅score (*q*, *di*​) +(1−*α*) ⋅math\_score (*q*, *di*​).

**Graph Representation of the mathematical expression**

-----------(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:

1. sqrt{}​ (square root operator)
2. ab (multiplication of a and b)
3. ≤ (less than or equal to)
4. \ (fraction operator)
5. a+b (addition)
6. 2(denominator)

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

1. **Establishing Edges (Relationships Between Nodes)**

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

​The square root node connects to ab.

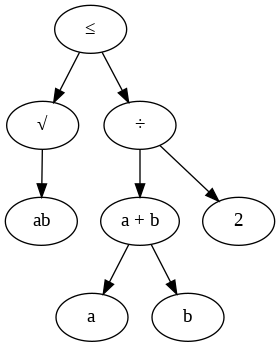
​ : The fraction node connects to the numerator (a+b) and denominator (2).

: The addition node connects to a and b.

**iv.** ≤ : The comparison operator connects to both sides of the inequality ( edge set is:

-----------(9)

1. **Graph Visualization**



Graph visualization of nodes

In Figure, 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.

**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=**---------**(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.

---------(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:

---------(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. ----------(13)

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

----------(14)

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