KDSH

2025

TEAM: DATA POOKIES

Table of Contents

1 PROBLEM STATEMENT

In academic and research domains, assessing the quality and publishability of **research papers** is crucial for advancing **knowledge** and **fostering innovation**. The process often **requires expert review**, which is **time-consuming** and subjective. With the rising volume of research outputs, there is a pressing **need for automated systems** to assist in evaluating the suitability of papers for publication.

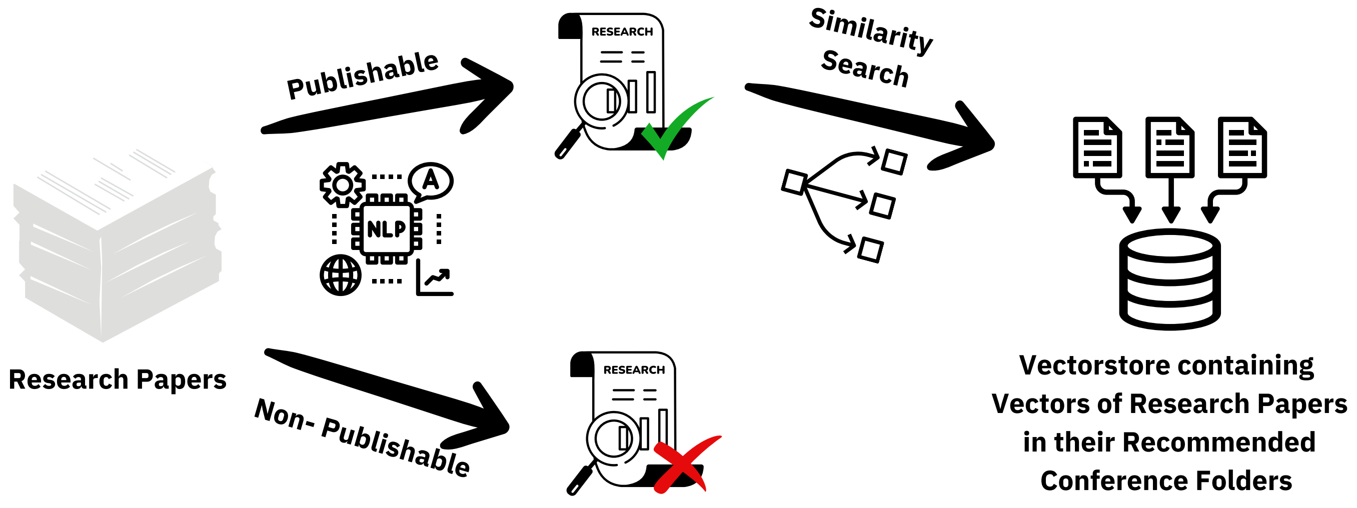
This task involves developing a **framework** to **classify** research papers as **Publishable** or **Non-Publishable** based on critical content evaluation. The framework must **identify issues** such as **inappropriate methodologies, incoherent arguments,** or **unsubstantiated claims** that compromise the paper's quality. For instance, papers lacking logical coherence, presenting unrealistic results without evidence, or applying **unsuitable methodologies** are categorized as **Non-Publishable**.

A dataset of **150 research papers**, including **15 labeled examples**, is provided to guide the **framework's development**. The solution must demonstrate **high accuracy**, **scalability**, and the ability to **maintain consistency** across diverse research domains. The outcome aims to streamline the **publication process** and **enhance objectivity** in research evaluation.

Selecting the most suitable **conference** for **submitting** a research paper is essential to ensure alignment with the **venue's scope and objectives**. This task involves creating a **framework** that evaluates a **paper's content**, **research focus**, and **methodology** to recommend appropriate conferences such as **CVPR, NeurIPS, EMNLP, DAA, TMLR,** or **KDD**.

The framework should compare the paper's characteristics against conference **profiles** and provide formal **justifications** for its **recommendations**. These justifications include a **comparative analysis** with reference papers to highlight the originality and significance of the research.

To enhance efficiency, the solution integrates **Pathway connectors** and utilize **Pathway Vectorstore** for **real-time data streaming** and **analysis**. The framework should classify only publishable papers and ensure they meet the **quality standards** and **relevance** of the selected **conference**. The recommendations must be accompanied by concise, evidence-backed rationales, ensuring **alignment** with academic excellence.


Publishable
RESEARCH
Similarity Search
NLP
Research Papers
Non- Publishable
RESEARCH
Vectorstore containing Vectors of Research Papers in their Recommended Conference Folders

2 DATASET DESCRIPTION

1. **Dataset Acquisition and Purpose**

- The dataset consist of **150 research papers**, with **15 labeled** as reference papers for classification guidance. Those 15 paper are further labeled with their **respective conference** name.

2. **Classification Labels**

- Papers are categorized into two labels: **Publishable** and **Non-Publishable**. The label is based on the **quality** of the paper, focusing on **methodology**, **coherence**, and **evidence of claims**.

3. **Dataset Format**

- The data is provided in a **pdf format** with fields such as **Paper ID** and **Label** (Publishable/Non-Publishable). The labeled papers are contained in the **folder** of the **conferences** they belong to.

4. **Data Cleaning and Outlier Handling**

The text preprocessing process involves:

- **Tokenization:** Splitting text into lowercase tokens.
- **Stop Word Removal:** Eliminating common, non-contributing words.
- **Non-Alphabetical Removal:** Keeping only alphabetic words.
- **Lemmatization:** Reducing words to their base form.

5. **Generating Tokens for the Data**

- The **tokenizer** converts **raw text** into tokens that the **model** can **process**. It **pads shorter sentences**, **truncates longer ones**, and ensures all inputs are **512** tokens long.

6. **Fine Tuning of LLM**

The code fine-tunes the **DistilBERT** model for sequence classification, using **Adam** optimizer and **sparse categorical cross-entropy loss**. The model is trained for **5 epochs** with **accuracy** as the **metric**, adapting it to classify research papers as Publishable or Non-Publishable.

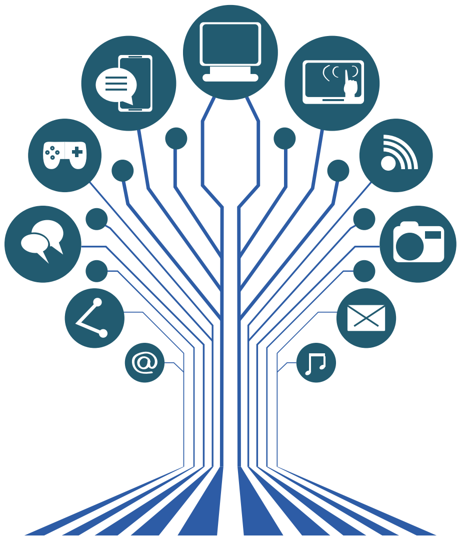
3 Data Preprocessing and Feature Engineering

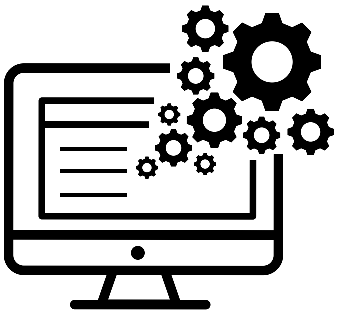
Data preprocessing is a critical step in **natural language processing (NLP) tasks**, especially when working with large text datasets. In this task, we have implemented several techniques to **clean**, **prepare**, and **tokenize** the research papers. Additionally, the dataset is **split** for **training** and **testing**, and features are engineered to **fit** the model input requirements.

1. **Text Preprocessing**:
   - **Tokenization**: Each paper's text is converted into tokens, transforming the sentences into smaller, **manageable units**.
   - **Lowercasing**: All text is converted to lowercase to ensure uniformity and **prevent case sensitivity** from affecting the model's **performance**.
   - **Stop Word Removal**: Common, **non-informative words** such as 'the', 'and', etc. are **removed** using the **NLTK** stop words list, **reducing** the **noise** in the data.
   - **Non-Alphabetical Token Removal**: Any token that contains **numbers or special characters** is **excluded**, focusing on meaningful words.
   - **Lemmatization**: Words are **reduced to their base** or **root form**, using a **WordNet lemmatizer** to ensure that words like 'running' and 'ran' are treated as 'run'.

2. **Dataset Splitting:**
   - **Training and Test Set Creation**: The labeled data (Publishable and Non-publishable papers) is split into training and testing sets, using an **80-20 ratio (80% for training and 20% for testing)**, which is a **standard practice** in machine learning to validate model performance.
   - **Random State**: The data split is **randomized** using a **fixed random seed** to ensure **reproducibility** of results.

3. **Tokenization and Data Preparation**:
   - **Tokenizer Selection**: The **Hugging Face AutoTokenizer** is used to tokenize the input text. The tokenizer is set to use the **distilbert-base-uncased** model, which is a **lightweight and efficient** version of **BERT**, known for its performance on sequence classification tasks.
   - **Padding and Truncation**: The texts are padded or truncated to a maximum length of **512 tokens**. This **standardizes** the **input size** and ensures that all input data **fits** the model's expected input dimensions.
   - **Return Tensors**: The tokenizer outputs are converted into **NumPy arrays**, which are compatible with **TensorFlow models** and can be efficiently processed.





4. **Feature Engineering:**
   - **Input Representation**: The text data is transformed into a format that the **model can understand** and process by converting the tokenized text into **embeddings** using the pre-trained tokenizer.
   - **Model Input Preparation**: The prepared data is now **ready** for feeding into the DistilBERT model for **fine-tuning** on the sequence classification task.

4 METHODOLOGY

Semi-Supervised Learning Approach for Research Paper Classification:

In this project, where only 15 labeled research papers are available for training, semi-supervised learning (SSL) plays a vital role in utilizing a significant amount of unlabeled data. By combining a small labeled dataset with a large unlabeled one, SSL enhances the model's generalization and performance. Below is an outline of the approach and itS key components.

1. Model Selection: TFDistilBERT for Sequence Classification

- A pre-trained TFDistilBERT model is used for sequence classification. This smaller and faster version of BERT benefits from transfer learning, leveraging prior training on extensive text corpora. Fine-tuning the "distilbert-base-uncased" model on the labeled dataset allows it to classify papers as "Publishable" or "Non-Publishable" based on observed patterns.

2. Model Compilation

- **Optimizer**: **Adam** optimizer, with a learning rate of 5e-5, ensures **efficient and gradual updates** to the model weights during training.
- **Loss Function**: **SparseCategoricalCrossentropy** is applied for its **effectiveness** in multi-class classification tasks.
- **Metric**: **Accuracy** is chosen as the evaluation metric for its simplicity and clarity in assessing classification performance.

3. Training the Model

- The model is trained for **five epochs** on the labeled data, gradually **minimizing the error**. The training dataset consists of papers labeled as "Publishable" or "Non-Publishable," providing the basis for learning.

4. Prediction on Unlabeled Data

- After training on labeled data, the model is used to classify unlabeled papers. **Predicted labels** (pseudo-labels) are generated and **added to the training dataset**, enabling the model to **learn** from both labeled and pseudo-labeled data.

5. Train-Test Split for Validation

- To validate the model, the labeled dataset is split into **training (80%)** and **testing (20%)** subsets. **Randomization** ensures **reproducibility**, allowing consistent evaluation of the model’s **generalization capability**.

6. Tokenization and Input Preparation

- The **Hugging Face AutoTokenizer** prepares the text data for processing. Input text is **tokenized**, **padded**, or **truncated** to a fixed length of 512 tokens for uniformity, and converted into **NumPy arrays** compatible with **TensorFlow**.

7. Data Augmentation Using Pseudo-Labels

- Predictions on **unlabeled papers** are used to expand the training dataset. This augmented dataset, comprising both labeled and pseudo-labeled data, is used to **retrain** the model, **enhancing its ability** to classify papers effectively.

8. Iterative Learning Process

- The semi-supervised approach involves iterative cycles of training, prediction, and retraining. With each **iteration**, the model **improves** its **accuracy** and **generalizes better**, even with a limited number of labeled examples.

9. Evaluation and Accuracy Metrics

- After training, the model is evaluated on the test set to assess its performance. **Accuracy** serves as the primary metric, reflecting the model’s ability to **classify papers correctly**.

10. Outcome

- This SSL approach demonstrates how limited labeled data can be effectively combined with unlabeled data to create a **robust classification model**. The iterative process ensures **improved performance**, **scalability**, and **adaptability**, making it well-suited for research paper classification tasks.

5 Model Selection and Training

TFDistilBertForSequenceClassification:

A **TFDistilBertForSequenceClassification** is a concise and efficient **natural language processing model**, leverages the power of transformers for **text classification** tasks. With its smaller size, pre-trained capabilities, and **seamless TensorFlow integration**, it offers a perfect balance of performance and **efficiency for real-world NLP applications**.

DistilBERT (Smaller, Faster Version of BERT): A **compressed version of BERT**, DistilBERT undergoes a distillation process to **reduce size and complexity** while retaining most of BERT's performance, making it **highly efficient** and **suitable for resource-constrained environments** like edge devices.

From_pretrained("distilbert-base-uncased"):
Loads a pre-trained model trained on large datasets, saving time and resources by leveraging a general language understanding that can be fine-tuned for specific tasks. It uses the distilbert-base-uncased variant, which includes:

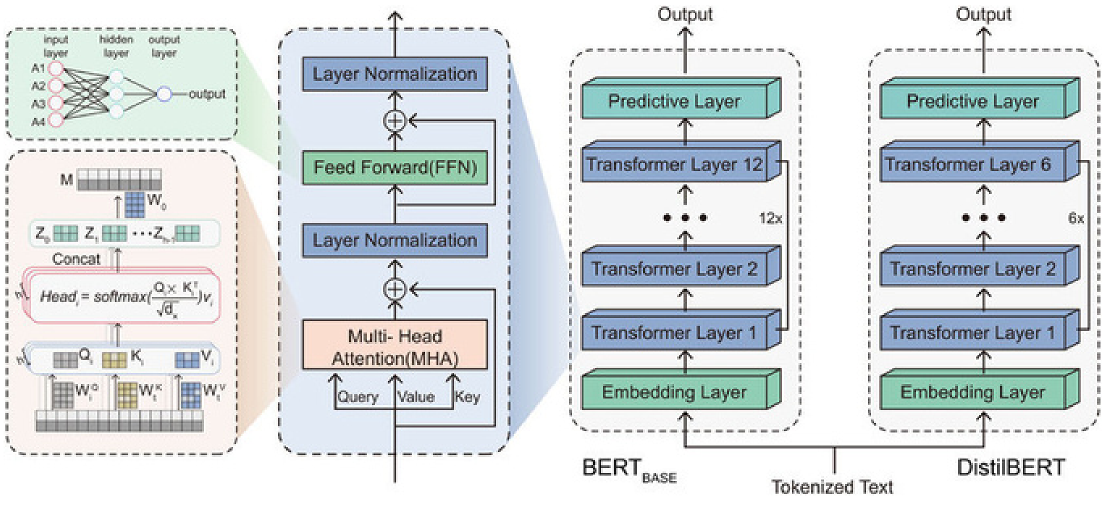
- **Pre-trained Model:** Trained on extensive corpora like Wikipedia for robust language comprehension.
- **distilbert-base-uncased:**
  - *Base*: Standard-size DistilBERT with 6 layers, 768 hidden units, and 12 attention heads.
  - *Uncased*: Processes text in lowercase, ignoring case sensitivity.

Training and Evaluation Metrics-

1. **Accuracy :**
   - **Accuracy measures the proportion of correctly predicted instances out of the total instances, reflecting the model's overall correctness.**

2. **F1 Score:**
   - **The F1 score is the harmonic mean of precision and recall, providing a balanced measure of a model's performance. It is particularly useful for evaluating classification tasks with imbalanced datasets.**

Architecture of DistilBERT



Results After and Before Applying Semi-Supervised learning Method -

- Clearly we saw that before Applying **Semi-Supervised learning** we are getting accuracy and F1 which are the sign of **Overfitting** because of less data but after applying this method we are getting good result with accuracy of 0.9 and F1 score of 0.9387

		ACCURACY	F1 SCORE
1	**Before**	1	1
2	**After**	0.9	0.9387

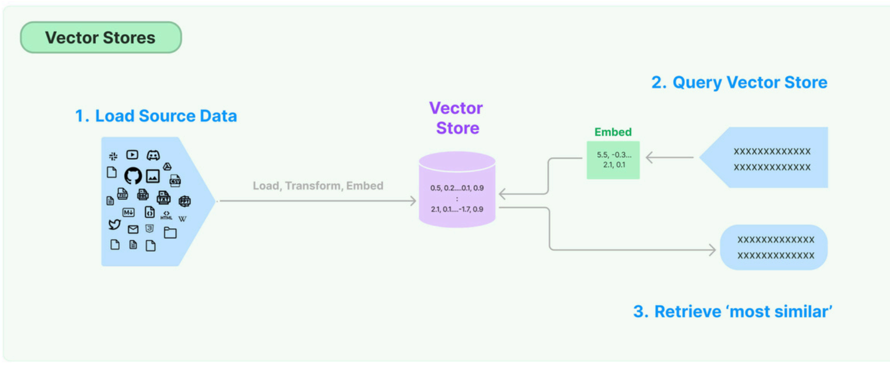
6 USING VECTOR STORES FOR DATA STORAGE AND SEARCH

This section describes the **implementation of a vector store system using Pathway** for efficient **document storage and similarity search capabilities.** The **system was designed to handle academic papers and provide fast, accurate retrieval of similar documents.**

A.Vector Store Architecture:

The implementation consists of two main components:

1. **Vector Store Server:** Handles document storage and embeddings.
2. **Vector Store Client**: Enables similarity searches across the document collection.



B.Server Implementation:

The vector store server was implemented using Pathway's VectorStoreServer class with the following key components:

- **Embeddings Model:** Utilized HuggingFace's "sentence-transformers/all-MiniLM-L6-v2" model for generating document embeddings
- **Text Splitting**: Implemented CharacterTextSplitter for breaking documents into manageable chunks
- **File System Connector:** Used Pathway's filesystem connector (pw.io.fs) to read documents in streaming mode
- **Caching System:** Implemented filesystem-based caching to improve query performance

C.Client Implementation:

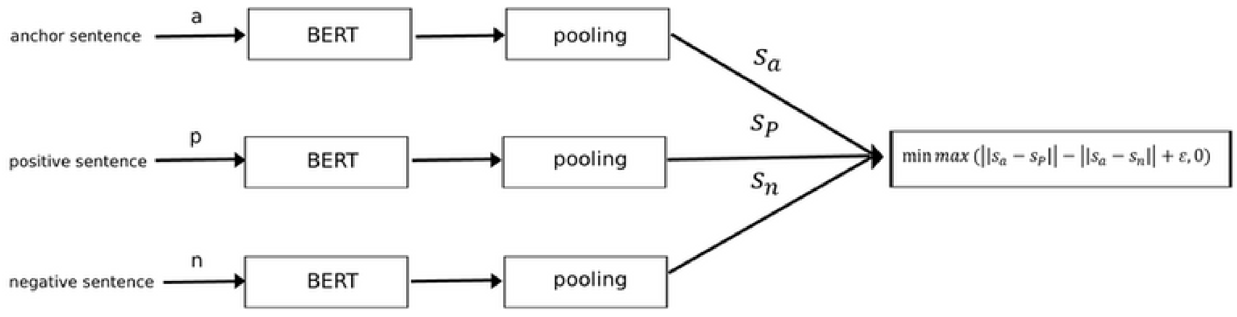
The client side was implemented using PathwayVectorClient, which provides an interface for performing similarity searches against the vector store:

client = PathwayVectorClient(host="127.0.0.1", port=8888)

Data Processing Pipeline

The system includes a comprehensive data processing pipeline that handles:

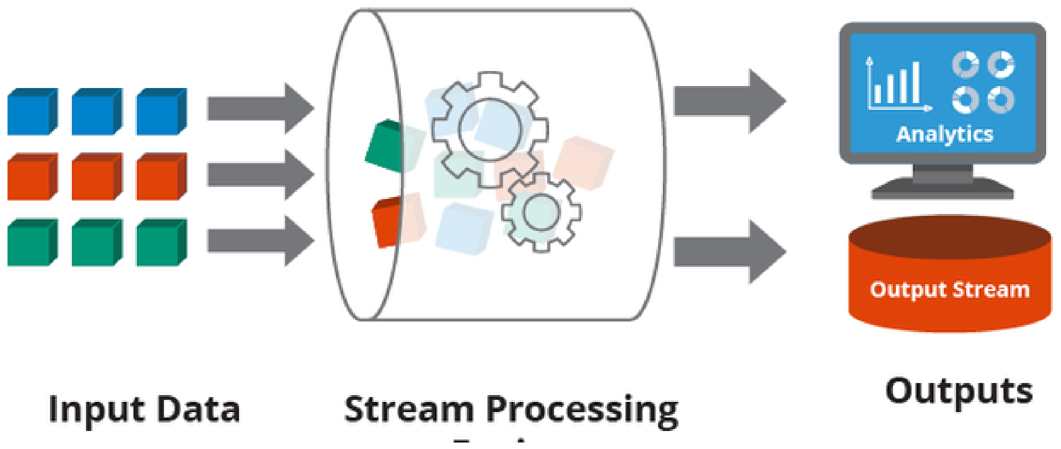
1. **Document Preprocessing**:
   - Text cleaning and normalization
   - Special character removal
   - Case normalization
2. **Feature Extraction:**
   - TF-IDF vectorization for keyword extraction
   - Automatic identification of key topics and themes
3. **Similarity Matching:**
   - Vector-based similarity search using cosine similarity
   - Results ranking and filtering



Performance Considerations:

The implementation includes several optimizations:

- **Streaming Mode:** Documents are processed in streaming mode to handle large datasets efficiently
- **Caching**: File system-based caching reduces redundant computations
- **Metadata Storage:** Document metadata is preserved for enhanced search results



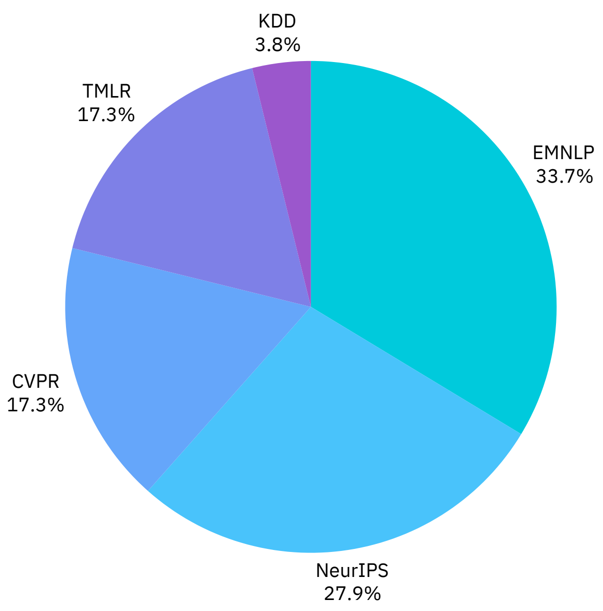
System Integration:

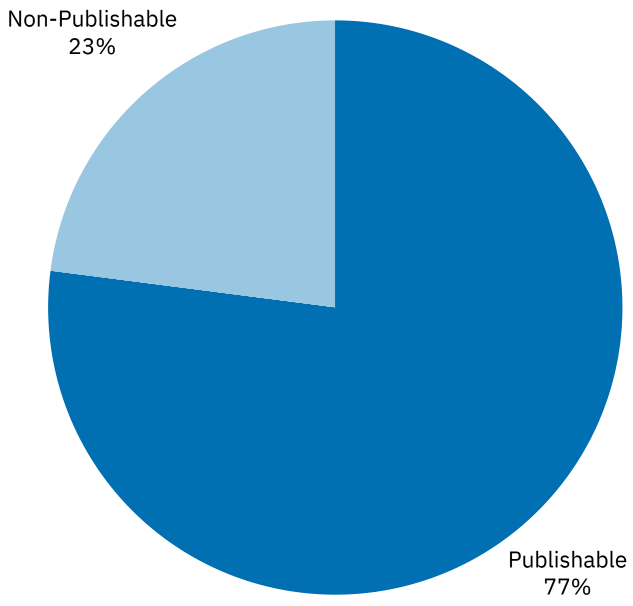
The vector store system integrates seamlessly with the larger document processing pipeline:

1. **Documents are first converted from PDF to text format**
2. **Text is preprocessed and embedded**
3. **Embeddings are stored in the vector store**
4. **Similarity searches can be performed through the client interface**

This architecture enables efficient storage and retrieval of academic papers while maintaining the context and relationships between documents.

6 RESULTS





1. Publishability Assessment
Out of all the research papers analyzed, **104 papers** were classified as **Publishable**, indicating they meet the **necessary standards for publication** based on the evaluation framework. These papers demonstrated coherence, methodological rigor, and evidence-based claims, qualifying them for further consideration in academic conferences.

2. Conference Recommendation
The classification framework successfully categorized the publishable papers into suitable conferences based on their content and **alignment with conference themes**. The distribution of papers across conferences is as follows:

- EMNLP: 35 papers
- NeurIPS: 29 papers
- CVPR: 18 papers
- TMLR: 18 papers
- KDD: 4 papers

Each classification was **accompanied by a well-reasoned rationale**, highlighting the **paper's alignment** with the **focus areas** and **standards** of the respective conference.

3. Key Observations

- **EMNLP** and **NeurIPS** emerged as the most recommended conferences, collectively accounting for more than half of the publishable papers. This suggests a significant focus on topics related to machine learning, natural language processing, and similar domains within the dataset.
- **CVPR** and **TMLR** received an equal number of recommendations, highlighting their relevance to papers with themes in computer vision and theoretical machine learning.
- **KDD** had the fewest papers recommended, likely reflecting a narrower focus on data mining themes within the dataset.

These results demonstrate the framework's capability to evaluate and classify research papers effectively, providing valuable insights for both researchers and organizers of academic conferences.

Annexure

Blogs Refered PAPERS REFERRED -

- https://pathway.com/blog/langchain-integration
- https://www.analyticsvidhya.com/blog/2024/07/tf-idf-matrix/
- https://huggingface.co/transformers/v3.0.2/model_doc/distilbert.html
- https://huggingface.co/docs/transformers/en/index
- https://medium.com/@tejaswaroop2310/tf-idf-understanding-the-power-of-text-representation-26582883410a