תמונה שמכילה טקסט, גופן, לוגו, גרפיקה

התיאור נוצר באופן אוטומטי

Software Engineering Department  
ORT Braude College

Capstone Project Phase A – 61998

**Metalanguage As an Interdisciplinary Classifier for Mathematics and Computer Science Fields**  
**24-2-R-10**

Ilya Vorotyntsev | Ilya.Vorotyntsev@e.braude.ac.il

Aleksandr Pitkin | Aleksandr.Pitkin@e.braude.ac.il

**Supervisors:**

Dr. Dan Lemberg | [lembergdan@braude.ac.il](mailto:lembergdan@braude.ac.il)

Mrs. Elena Kramer | [elenak@braude.ac.il](mailto:elenak@braude.ac.il)

**Table Of Contents**

[Abstract 3](#_Toc177852197)

[1. Introduction 3](#_Toc177852198)

[2. Literature Review 4](#_Toc177852199)

[2.1 Traditional Approaches to Text Classification 5](#_Toc177852200)

[2.2 Evolution to Deep Learning Techniques 5](#_Toc177852201)

[2.3 Applications in Educational Text Classification 6](#_Toc177852202)

[2.4 Traditional Approaches to Text Clustering 6](#_Toc177852203)

[3. Background 7](#_Toc177852204)

[3.1 XLNet 7](#_Toc177852205)

[3.2 K-Means Clustering 11](#_Toc177852206)

[3.3 PAM (Partitioning Around Medoids) 11](#_Toc177852207)

[3.4 Density-Based Clustering 12](#_Toc177852208)

[3.5 Gaussian Mixture Models (GMMs) 13](#_Toc177852209)

[4. Expected Achievements 13](#_Toc177852210)

[5. Proposed Work 14](#_Toc177852211)

[5.1 Dataset Collection 14](#_Toc177852212)

[5.2 Text Preprocessing 15](#_Toc177852213)

[5.3 Text Classification 16](#_Toc177852214)

[5.4 Text Clustering 17](#_Toc177852215)

[6. Evaluation metric 18](#_Toc177852216)

[6.1 Text Classification Metrics 19](#_Toc177852217)

[6.2 Text Clustering Metrics 20](#_Toc177852218)

[7. Challenges 21](#_Toc177852219)

[7.1 Dataset Challenges 21](#_Toc177852220)

[7.2 Preprocessing Difficulties 22](#_Toc177852221)

[7.3 Model Training and Hyperparameter Optimization 22](#_Toc177852222)

[7.4 Classification and Clustering without Domain-Specific Terms 22](#_Toc177852223)

[8. Testing Plan 23](#_Toc177852224)

[9. Bibliography 24](#_Toc177852225)

# Abstract

Providing students with the most effective learning resources is essential to improving educational outcomes, particularly in fields like mathematics and computer science, which involve complex and varied concepts. To better understand how these subjects are taught, this capstone project explores the linguistic structures that underlie educational texts across specific fields of mathematics and computer science.

Our methodology involves the use of advanced natural language processing (NLP) techniques, such as stemming, lemmatization, and tokenization, to strip academic texts of domain-specific terminology. What remains is the metalanguage—a more general form of language that consists of common words and structures used to communicate ideas across different fields.

We analyze these preprocessed texts using XLNet, a deep learning model, to classify them into their respective fields. In addition to classification, we apply clustering techniques (K-means, PAM, Density Clustering, and Gaussian Mixture) to verify the classification results. This dual approach allows us to identify correlations between classification accuracy and the natural grouping of texts, offering insights into whether texts from related fields share similar linguistic structures.

By successfully classifying and clustering educational texts without relying on subject-specific terminology, this research contributes to the development of tools that can better adapt educational materials to the diverse needs of students.

**Keywords:**

Text Classification, Natural Language Processing, Clustering, Mathematics and Computer Science Education.

# Introduction

Students often face different challenges in various academic fields. Although mathematics and computer science share logical foundations, they require distinct skills, leading some students to excel in one area while struggling in the other. This imbalance can result in lower performance or even students dropping out of courses. The common one-size-fits-all teaching approach does not always address these differences, making it harder for students with different strengths and weaknesses to succeed. As a result, there is a growing need for learning strategies that better match individual student needs.

This project was inspired by the observation that students often face difficulties in specific fields within mathematics or computer science. Our aim is to explore the differences in the language used across various fields and to classify educational texts accordingly. We focus on fields in mathematics, such as Linear Algebra, Abstract Algebra, Calculus, Combinatorics, Probability, Logic, and Set Theory, and fields in computer science, such as Data Structures and Algorithms, Object-Oriented Programming (OOP), Imperative Programming, Functional Programming, Operating Systems, Compilation, and Automata and Formal Languages.

Traditional methods of text classification in education often rely on subject-specific terminology to distinguish between fields. Approaches like keyword analysis and topic modeling are useful but limited, as they depend heavily on these terms and do not explore the deeper linguistic patterns that may be shared across different fields.

In this project, we aim to go beyond traditional methods. By removing domain-specific terminology from educational texts, we focus on the remaining general linguistic structures, or metalanguage, that are common across different fields. Our goal is to identify patterns in this metalanguage that can help classify texts from different fields of mathematics and computer science.

To achieve this, we apply natural language processing (NLP) techniques, such as stemming, lemmatization, and tokenization, using tools like NLTK. After preprocessing, we convert the texts into vectors using methods like doc2vec, which capture the underlying meaning of the text. We then use XLNet [1], a deep learning model, to classify the texts based on these patterns. Additionally, we apply clustering techniques, including K-means, PAM, Density Clustering, and Gaussian Mixture, to verify the classification results [2] [3]. This approach helps us check if related fields are grouped together in the same cluster.

This research can make a significant contribution to educational technology by enabling the development of tools that help students transition more easily between different fields. These tools could present educational material in a way that better fits each student's strengths and weaknesses. In the following sections, we will explore existing work on educational text classification, outline the background and methodology of this project, and discuss how we will evaluate and verify our findings.

# Literature Review

This section reviews the evolution of text classification methodologies, emphasizing the transition from traditional approaches to modern deep learning techniques. In addition to text classification, text clustering methods are essential for grouping similar texts without prior labels. Techniques such as K-means, Partitioning Around Medoids (PAM), Density-based clustering, and Gaussian Mixture models are widely used in natural language processing to uncover hidden patterns within large datasets [3].

## Traditional Approaches to Text Classification

Historically, text classification relied heavily on machine learning algorithms such as Naive Bayes, Support Vector Machines (SVM) [4], and decision trees. These models required extensive feature engineering, where text features such as word frequencies, TF-IDF scores, and n-grams were manually extracted and fed into the classifiers [5]​. Despite their effectiveness, these traditional approaches often struggled with the complexities of language, especially in handling nuances like context, polysemy, and semantic relationships.

**Naive Bayes** classifiers, one of the earliest models used for text classification [4], operate on the assumption that features are independent given the class label. While this assumption is rarely true in practice, Naive Bayes models perform surprisingly well, especially in scenarios where the independence assumption approximately holds, such as spam detection [6]​.

**Support Vector Machines (SVMs)**, another popular traditional method, utilize a hyperplane to separate classes in a high-dimensional feature space. SVMs are particularly effective for binary classification tasks and are known for their robustness in handling high-dimensional data. However, they require careful tuning of hyperparameters and kernel functions to perform optimally [5].

Despite their success, these traditional approaches have limitations, particularly in their ability to capture the deeper semantics of the text. They are heavily reliant on the quality of the input features and often require large amounts of labeled data to perform well. Additionally, they may not generalize effectively to unseen data or complex language patterns.

## Evolution to Deep Learning Techniques

The advent of deep learning has revolutionized text classification, enabling models to automatically learn features from raw text data. **Convolutional Neural Networks (CNNs)** and **Recurrent Neural Networks (RNNs)** are among the earliest deep learning architectures applied to text classification tasks.

**CNN**s were initially developed for image processing but have proven effective in NLP tasks due to their ability to capture local dependencies in text. By applying convolutional filters over word embeddings, CNNs can identify significant phrases or n-grams, which are crucial for classification [6] [5].

**RNN**s, particularly Long Short-Term Memory (LSTM) networks, are designed to capture sequential dependencies in text. LSTMs address the vanishing gradient problem in traditional RNNs, allowing them to learn long-term dependencies, making them well-suited for tasks where the order of words is important [4] [6]​. However, RNNs can be computationally expensive and are often slower to train compared to CNNs [4].

The introduction of **Attention Mechanisms** and **Transformers** has further advanced the field. Transformers, such as the XLNet [1]. model, have become the state-of-the-art in text classification. Unlike RNNs, Transformers do not process text sequentially; instead, they rely on self-attention mechanisms to weigh the importance of each word in a sentence relative to all other words. This allows them to capture contextual information more effectively, leading to significant improvements in classification accuracy [5].

## Applications in Educational Text Classification

In educational contexts, text classification can be used to tailor learning materials to students' needs by automatically categorizing content based on difficulty level, topic, or other relevant criteria. For example, using semantic networks [6] can enhance the understanding of educational texts by representing relationships between concepts in a graph-like structure, making it easier for machine learning models to classify and generate educational content [6].

Various AI techniques such as Artificial Neural Networks (ANNs), Classification and Regression Trees (CARTs), and decision trees were evaluated for their ability to classify educational texts based on their knowledge content. The study [5] found that ANNs performed significantly better in distinguishing between texts designed to transfer knowledge and those that do not [5]. This highlights the potential of advanced AI techniques in improving educational outcomes through more precise text classification.

## Traditional Approaches to Text Clustering

Clustering techniques are widely used in data analysis to uncover hidden structures and patterns within large datasets, offering various methods to group similar data points. K-means, one of the most widely used clustering methods, partitions data into *k* clusters by minimizing the squared Euclidean distance between points and their respective cluster centroids. Although it is computationally efficient, K-means requires prior knowledge of the number of clusters and is sensitive to initial centroid placement [3]. Partitioning Around Medoids (PAM), a more robust alternative, uses actual data points (medoids) to represent clusters, making it less sensitive to outliers. PAM begins with an initial set of medoids and iteratively improves the clustering by swapping medoids with non-medoids to reduce the overall distance within clusters, though it can be computationally expensive [2]. Density-based clustering, such as DBSCAN, identifies clusters based on regions of high data density, making it particularly useful for discovering clusters of arbitrary shapes without needing to predefine the number of clusters [3]. Lastly, Gaussian Mixture Models (GMMs) take a probabilistic approach by assuming that the data is generated from a mixture of Gaussian distributions. GMMs allow for soft clustering, where each data point can belong to multiple clusters with varying probabilities, making them effective for handling overlapping clusters [3]. Each of these methods offers distinct advantages, with their application depending on the nature of the dataset and the clustering objectives.

# Background

Text classification is a crucial task in natural language processing (NLP) that has evolved with the development of advanced machine learning models designed to handle the complexities of language. For this project, we focus on XLNet, a state-of-the-art model known for its performance in various text classification tasks [7] [8] [9]. In this work, XLNet will be used to classify texts from different fields within mathematics and computer science, while clustering techniques like K-means, PAM, Density Clustering, and Gaussian Mixture will be applied to further analyze the patterns in the classified texts.

## XLNet

XLNet is a powerful model developed for language understanding tasks, built upon the foundation of the Transformer-XL architecture. Unlike traditional models that process text sequentially, XLNet [1] analyzes all possible word arrangements (or permutations) in a sentence, allowing it to capture a broader context and develop a deeper understanding of language. This permutation-based approach enables XLNet to grasp complex language patterns that other models might miss, making it particularly effective in tasks such as text classification and document understanding [10].

* Permutation-based Training

A key innovation in XLNet is its permutation-based training strategy [Figure 1]. Traditional language models predict words in a fixed left-to-right or right-to-left order, which limits the model's ability to capture the full context of a sentence. XLNet, on the other hand, generates all possible permutations of the input sequence during training. This allows the model to predict words based on a richer context, where each word is influenced by all other words in different arrangements. By doing this, XLNet can better understand long-range dependencies and capture complex relationships between words, resulting in more accurate predictions [10].

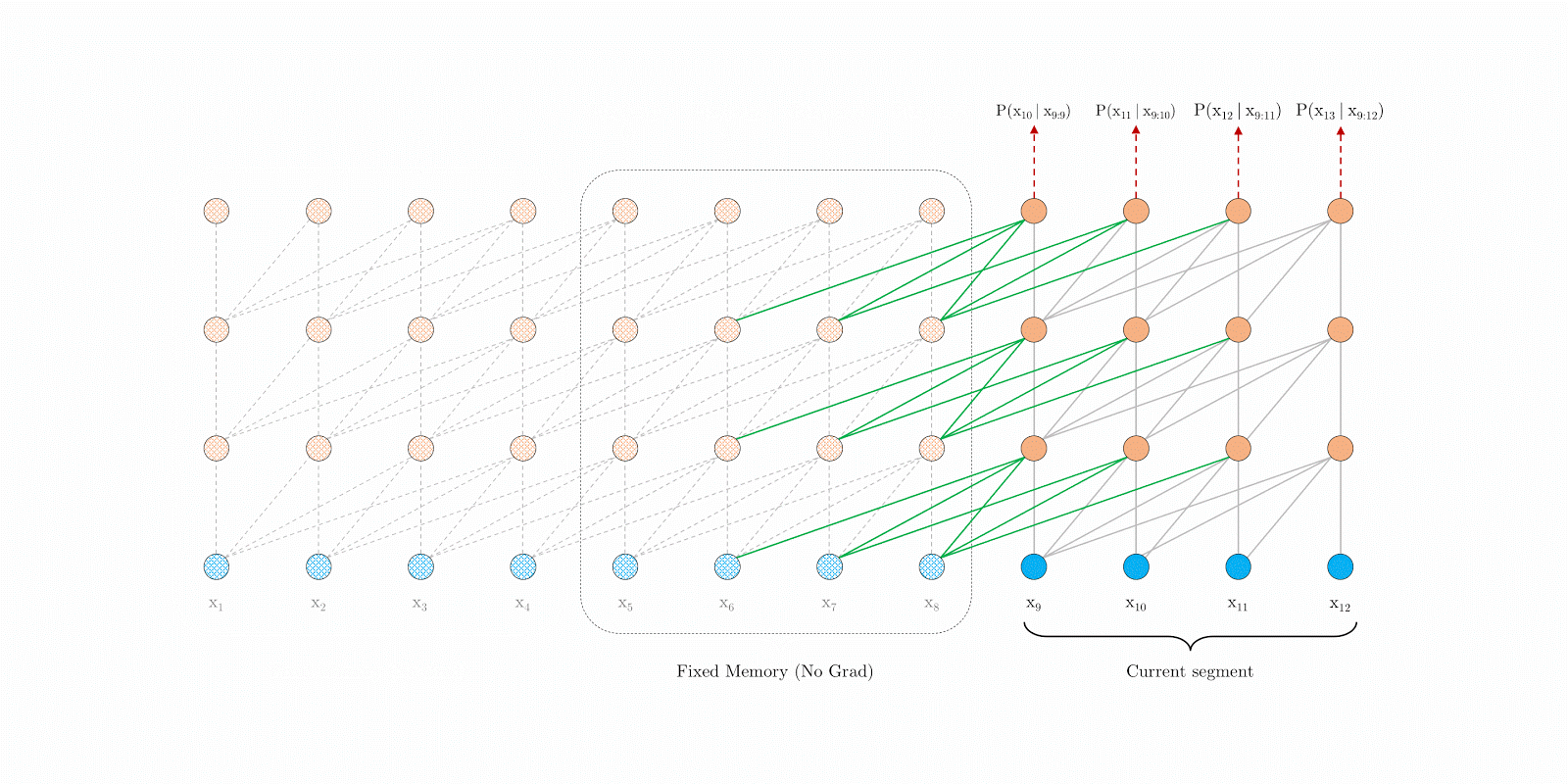


Figure 1: Permutation-based approach of XLNet, illustrating how the model processes different word arrangements to better capture contextual relationships within the text.

* Attention Masking

While processing permutations, XLNet uses attention masking to maintain the correct word order and ensure that each word is aware of its surrounding context. Attention masking allows the model to control what information is visible to each word during the learning process. For instance, if a word appears in the middle of a sentence, the attention mask will only allow XLNet to consider the relevant words in its context while ignoring others. This ensures that even though the model is processing different word permutations, it still retains the original sentence structure and meaning.

* Two-Stream Attention Mechanism

XLNet introduces a two-stream attention mechanism, which is designed to improve how the model predicts the next word in a sequence. In this setup, XLNet maintains two separate streams of information: one stream (the content stream) focuses on the actual content of the input tokens, while the other stream (the query stream) is used to predict the next word based on its position in the sentence. This architecture enables XLNet to handle bidirectional contexts more effectively, as the model is able to make position-aware predictions without knowing whether a word belongs to the sentence, improving its ability to generalize across different tasks. [Figure 3] presents an overview of the proposed permutation language modeling with two-stream attention [10].

* Segment Recurrence

One of the standout features of XLNet is its segment recurrence mechanism [Figure 2], borrowed from Transformer-XL, which allows the model to handle long sequences of text more efficiently. In traditional Transformer models, each segment of text is processed independently, which limits the model’s ability to retain information from previous segments. XLNet overcomes this limitation by reusing the hidden states from previous segments, allowing it to maintain a continuous flow of information across longer texts. This feature is particularly useful for processing documents that span multiple paragraphs, where understanding the entire context is critical.

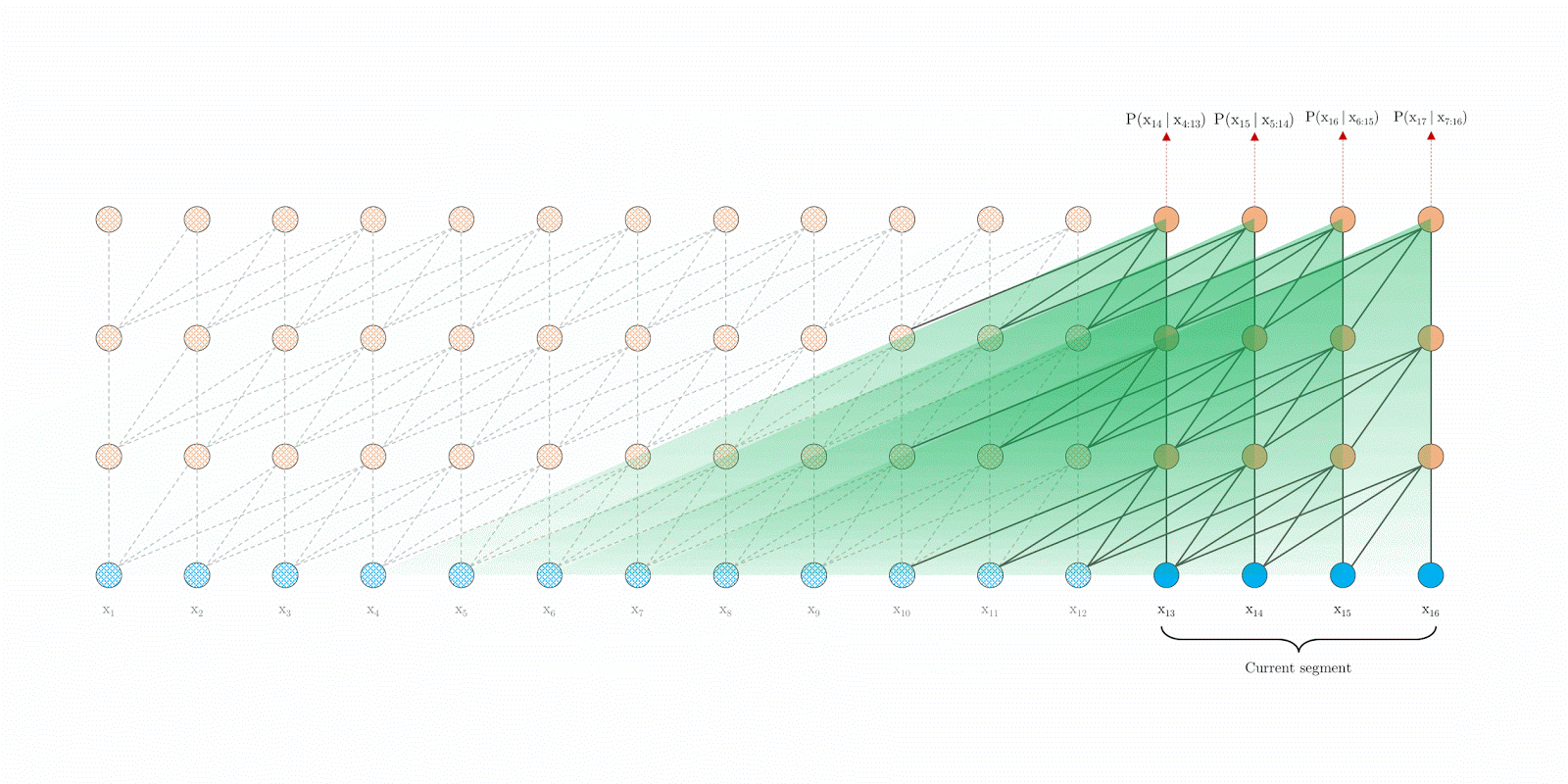


Figure 2: Segment recurrence mechanism in XLNet, showing how information from previous text segments is reused to enhance understanding of longer text sequences.

* Relative Positional Encoding

XLNet also improves upon traditional models with its use of relative positional encoding. Instead of relying on the absolute positions of words in a sentence, which can vary widely between different texts, XLNet focuses on the relative positions of words to one another. This allows the model to better understand how words relate within a given context, regardless of their specific order in the text. By focusing on the relative distance between words, XLNet can capture more flexible sentence structures, making it better suited for handling complex, non-linear sentence patterns.

* Autoregressive Objective

XLNet combines the best of autoregressive models and bidirectional models by using a generalized autoregressive training objective. This allows the model to generate word predictions based on both left-to-right and right-to-left contexts, providing a more complete understanding of each word’s role in the sentence. This autoregressive approach is a key factor behind XLNet’s ability to outperform many traditional models, as it can capture dependencies between distant words while still maintaining bidirectional context [10].

* Pretraining and Fine-tuning

Finally, like other state-of-the-art language models, XLNet undergoes two phases: pretraining and fine-tuning. During pretraining, the model is exposed to a large corpus of text data where it learns general language patterns using its permutation-based approach. After pretraining, XLNet can be fine-tuned for specific tasks such as text classification, question answering, or sentiment analysis. The flexibility of the model allows it to adapt to different tasks while maintaining high performance across various language-related benchmarks.

A diagram of a medical procedure

Description automatically generated with medium confidence

Figure 3: (a): Content stream attention, which is the same as the standard self-attention. (b): Query stream attention, which does not have access information about the content . (c): Overview of the permutation language modeling training with two-stream attention.

Overall, XLNet’s innovative architecture—built around permutation-based training, attention masking, and segment recurrence—makes it one of the most advanced models for natural language processing. Its ability to handle long-range dependencies, predict word order in both directions, and retain contextual information across lengthy texts gives it a significant advantage in tasks like text classification, question answering, and document understanding.

## K-Means Clustering

K-means is one of the most widely used clustering methods due to its simplicity and solid mathematical foundation. The core idea is to partition a dataset into *k* clusters, where each cluster is represented by its centroid, which is the arithmetic mean of the data points within that cluster. The goal is to minimize the total squared error, which represents the deviation of the data points from their respective cluster centroids. This is achieved by iteratively assigning data points to the nearest centroid and recalculating the centroids based on the updated clusters. The algorithm [Figure 4] continues until convergence, where the positions of the centroids and the cluster assignments no longer change. While easy to implement and computationally efficient, K-means has some limitations, such as sensitivity to the initial placement of centroids and the need to specify the number of clusters beforehand.

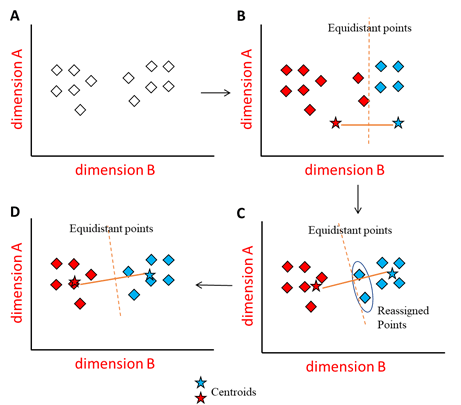


Figure 4: This illustration outlines the steps of the K-means clustering algorithm, including the initial random placement of centroids (Panel A), the assignment of points to centroids (Panel B), the adjustment of centroids based on their clusters (Panel C), and the final stabilization of clusters (Panel D).

## PAM (Partitioning Around Medoids)

PAM is a clustering technique that improves upon K-means by using actual data points, called "medoids," to represent clusters. A medoid is the most centrally located data point within a cluster, making it more representative and robust to outliers than centroids, which are average values in K-means. The process starts with the BUILD algorithm, which selects an initial set of medoids by iteratively choosing points that minimize the total distance between them and all other data points. After this initialization, the SWAP algorithm is used to refine the clusters by swapping medoids with non-medoids to further reduce the total distance within the clusters. This iterative process continues until no further improvements can be made. While PAM is more accurate and resistant to outliers compared to K-means, it is computationally expensive, requiring O(n²) time for distance calculations, which limits its scalability for large datasets.

## Density-Based Clustering

Density-based clustering identifies clusters based on regions of high object density, separated by areas of low density. This approach assumes that data points are sampled from an unknown underlying probability distribution, and clusters are defined as dense regions of the data space. The most well-known density-based clustering algorithm is DBSCAN, which groups points that are closely packed together and marks points in low-density regions as outliers. Unlike methods such as K-means [Figure 5], density-based clustering does not require specifying the number of clusters in advance and can handle clusters of arbitrary shapes and sizes. The flexibility of this approach makes it particularly useful for identifying complex structures in data. Density-based methods are also robust to noise and can effectively detect outliers.

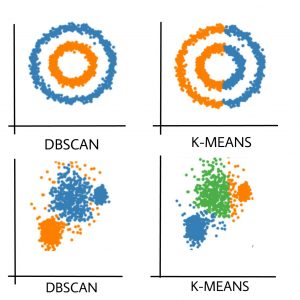


Figure 5: DBSCAN accurately captures the circular patterns and identifies distinct clusters in dense areas, whereas K-means segments the data into pie slices and arbitrary groups, failing to conform to the natural data shapes.

## Gaussian Mixture Models (GMMs)

Gaussian Mixture Models (GMMs) approach clustering by assuming that data points are generated from a mixture of several Gaussian distributions. A Gaussian distribution, or normal distribution, is a bell-shaped curve that represents how data points are distributed around a mean, with most points close to the mean and fewer points as you move further away. In GMMs, each Gaussian distribution represents a cluster, and data points are assigned to these clusters based on the probability of belonging to each distribution. These probabilities come from how closely a data point matches the characteristics (mean and variance) of each Gaussian distribution.

To fit the model, GMMs use the Expectation-Maximization (EM) algorithm, which finds the parameters (mean and variance) of each Gaussian distribution. EM iteratively adjusts these parameters to improve the match between the model and the data until it converges on the best possible fit. GMMs are particularly effective when the data follows a normal distribution, and they work well with overlapping clusters, which methods like K-means struggle to handle. GMMs also allow for soft clustering, meaning that each data point can belong to multiple clusters with varying probabilities, rather than being assigned to just one cluster. However, the method relies on knowing the number of distributions (or clusters) in advance, and its performance can degrade if the assumed distribution does not match the data well.

# Expected Achievements

In this project, our primary goal is to distinguish between metalanguages of various fields within mathematics and computer science, even after removing all domain-specific terminology.

We anticipate that XLNet will classify texts from these distinct fields with a moderate level of accuracy. If the model achieves very high accuracy (e.g., over 85%), it could suggest that the metalanguages of each field are entirely unique, showing little similarity between them. However, some fields in mathematics and computer science share similar metalanguage, such high accuracy might also point to potential issues with our data or model. Conversely, if the model achieves low accuracy, this could indicate problems with the model or data, or that the metalanguages of different fields are so similar that the model cannot effectively distinguish between them. Therefore, we expect to observe an intermediate level of accuracy, reflecting both the uniqueness and the overlaps in metalanguage across the fields. This would suggest that while there are differences in language use, structure, and context, there are also significant similarities that challenge the model's ability to distinguish between fields entirely.

A key component of this project is the use of clustering techniques, such as K-means, PAM, Density Clustering, and Gaussian Mixture, to verify the classification results from XLNet. After classification, we expect that related fields—especially those with overlapping linguistic structures—will cluster together. For instance, if XLNet struggles to differentiate between texts from closely related fields, such as Calculus and Probability, we expect clustering analysis to reveal that these texts naturally group together, offering a deeper understanding of their linguistic similarities.

In addition, we will explore how various hyperparameters, such as learning rate, batch size, sequence length, and the number of layers in XLNet, influence the model's performance. Fine-tuning these hyperparameters will be critical to optimizing the model’s accuracy and efficiency. We anticipate that adjustments in these parameters will significantly impact how well XLNet captures the nuances of different fields, as the removal of domain-specific terms forces the model to rely on more subtle linguistic structures.

Moreover, the project will evaluate the effectiveness of the different clustering techniques in verifying classification results. Each method offers distinct advantages: K-means is efficient for large datasets, PAM is more robust to outliers, Density Clustering can handle non-linear clusters, and Gaussian Mixture Models (GMMs) allow for soft clustering, where texts can belong to multiple clusters with varying probabilities. Through this comparative analysis, we aim to identify which clustering technique best complements XLNet’s classification and provides the most meaningful insights into the underlying linguistic patterns of the texts.

Ultimately, the success of this project could have far-reaching implications for designing personalized educational materials. By understanding how metalanguage functions across different fields, we can create adaptive learning resources that cater to students’ individual strengths and weaknesses, offering more accessible ways to learn mathematics and computer science.

# Proposed Work

This section describes the processes for **Dataset Collection**, **Text Preprocessing**, **Text Classification and Text Clustering**, including the models and techniques we will examine.

## Dataset Collection

The dataset for this research was manually compiled from publicly available sources on the internet, with a focus on freely accessible academic books in the fields of mathematics and computer science. To ensure the relevance and quality of the data, each book was individually reviewed to confirm its suitability for our research objectives. The dataset includes both original PDF files and scanned PDFs, where the latter contain images of book pages rather than digitally embedded text.

For the extraction of text from the original PDF documents, we will experiment with several text extraction algorithms, including PyMuPDF (Fitz), pdfminer.six, and PyPDF2. These tools differ in their capabilities to handle complex PDF structures, such as embedded fonts, non-standard character encoding, and multiple layers of metadata. A comparative evaluation will be conducted to determine which tool provides the most accurate and reliable text extraction for our dataset. In the case of scanned PDFs, where the text is rendered as an image, we will employ **Tesseract**, an optical character recognition (OCR) engine, to convert the image data into machine-readable text.

## Text Preprocessing

The preprocessing phase is critical for ensuring that our dataset is appropriately structured for further analysis and model training. The first step involves removing irrelevant pages from the text, such as bibliographies, tables of contents, indices, and other sections that do not directly pertain to the book’s subject matter. We aim to develop an algorithm capable of predicting the number of pages to be excluded from each document. In the event that this approach proves ineffective, a fallback strategy utilizing predefined, hard-coded values will be implemented.

Once the dataset has been cleansed of irrelevant pages, we will apply a series of standard preprocessing techniques to prepare the text for analysis. These include converting all characters to lowercase, removing punctuation marks, and eliminating extra whitespace. Additionally, numerical values will be removed, leaving only alphabetic characters and spaces. This process will ensure that the dataset consists solely of textual data relevant to the task at hand

As a result of the text extraction from PDF files, some irrelevant characters may be introduced into the dataset. To address this, we will implement a filtering process using the **wordfreq** library, which will help eliminate non-standard words by ensuring that only terms of a minimum frequency and at least two characters in length are retained. This step is vital to improving the quality of the dataset and minimizing the presence of noisy or irrelevant data

Next, we will employ natural language processing techniques using the **Natural Language Toolkit (NLTK)** to perform stemming and lemmatization. These processes will reduce words to their root forms and standardize different word variations, enabling the identification of common linguistic structures without being skewed by word inflections.

To ensure that all domain-specific terminology is effectively removed from the text, we will utilize a manually curated dictionary of subject-specific terms. This dictionary will contain terms of varying lengths—one-word, two-word, and three-word phrases—reflecting the complexity of terminology in mathematical and computer science texts. A specialized algorithm will be employed to identify and remove these terms from the dataset, ensuring that the texts are devoid of domain-specific language and leaving only general linguistic patterns.

Following this, we will experiment with two different approaches to stop word removal. The first approach will remove common English stop words, while the second approach will retain these words. We will evaluate both approaches to determine which yields better classification results, as the presence or absence of stop words may influence the underlying linguistic patterns in the texts.

Finally, the preprocessed text will be segmented into chunks of varying sizes. These segments will serve as input for the deep learning models. We will explore different segment sizes and empirically select the size that delivers the most accurate classification results in subsequent stages of the project.

## Text Classification

To classify educational texts in the fields of mathematics and computer science after stripping away domain-specific terminology, we will employ XLNet, a state-of-the-art deep learning model for text classification. XLNet’s autoregressive framework and permutation-based training enable it to capture complex linguistic patterns and bidirectional dependencies [10], making it well-suited for our task. The following parameters and optimization techniques will be used to enhance XLNet’s performance:

* Sequence Length:

The choice of sequence length plays a significant role in the model’s ability to capture meaningful context. We will experiment with varying sequence lengths, aiming to balance computational efficiency with the model’s ability to capture long-term dependencies in texts that span multiple pages. Texts from fields like Calculus or Automata Theory may require longer sequences to capture complete context, while fields with shorter explanations, such as Logic, may benefit from shorter sequences.

* Batch Size:

We will experiment with different batch sizes to find the optimal balance between training speed and memory usage. A smaller batch size may improve generalization, while a larger batch size could accelerate training.

* Learning Rate:

A critical parameter for optimizing XLNet, the learning rate will be fine-tuned through a grid search or other optimization techniques. We will start with a smaller learning rate to prevent overshooting and gradually increase it, evaluating how well the model converges.

* Regularization Techniques:

To avoid overfitting, we will incorporate regularization techniques such as dropout and weight decay. Dropout will help reduce reliance on specific neurons during training, while weight decay will penalize large weights, forcing the model to generalize better.

By carefully tuning these parameters and utilizing various optimization techniques, we aim to maximize XLNet’s accuracy in classifying educational texts.

* 1. Text Clustering

In addition to classification, clustering techniques will be employed to validate the results from XLNet and further explore relationships between fields. Below are the parameters and techniques we will utilize to optimize clustering performance based on our specific dataset:

* K-means:
  + Number of Clusters (k): We will use the "elbow method" and the silhouette score to determine the optimal number of clusters. Both techniques help in identifying the value of *k* where adding more clusters doesn’t significantly reduce within-cluster variance.
  + Initialization (k-means++): We will employ the k-means++ initialization technique to ensure that the initial centroids are well distributed, reducing the likelihood of poor clustering results.
  + Maximum Iterations and Convergence: The number of iterations will be optimized to ensure convergence without excessive computation, setting a balance between accuracy and performance.
* PAM (Partitioning Around Medoids):
  + Initial Medoid Selection (BUILD Algorithm): We will use the BUILD algorithm to carefully choose the initial medoids, ensuring that we start with a strong initial configuration.
  + Optimization Iterations (SWAP Algorithm): The SWAP algorithm will be iteratively applied to refine the medoid positions. We will experiment with different iteration limits to avoid overfitting while maximizing intra-cluster homogeneity.
  + Dissimilarity Measure: Instead of the default Euclidean distance, we will evaluate other dissimilarity measures (such as Manhattan distance) to determine which one best suit the structure of our text data.
* Density Clustering (e.g., DBSCAN):
  + Epsilon (ε): We will experiment with the epsilon parameter, which determines the maximum distance between two points for them to be considered in the same neighbourhood. By running multiple trials with different values of ε, we aim to identify a balance between finding well-separated clusters and avoiding excessive noise.
  + MinPts (Minimum Points): The MinPts parameter (minimum number of points to form a dense region) will be adjusted to fit the scale of our dataset. Larger values of MinPts may help avoid small, noisy clusters, while smaller values may reveal more granular patterns.
  + Distance Metric: The performance of DBSCAN will be tested with various distance metrics (e.g., cosine similarity, which is useful for text data) to identify clusters based on more context-aware measures of similarity.
* Gaussian Mixture Models (GMMs):
  + Number of Components: We will use the Bayesian Information Criterion (BIC) and the Akaike Information Criterion (AIC) to choose the optimal number of Gaussian components (i.e., clusters) to prevent overfitting or underfitting.
  + Covariance Matrix Type: We will experiment with different covariance matrix settings (full, tied, diagonal, and spherical) to determine how well each setting models the data, especially given the potential overlap between fields.
  + Initialization: GMMs will be initialized using k-means clustering, ensuring that the initial parameters for the Gaussian distributions are well-aligned with the data.
  + Regularization Parameter: A regularization term will be added to the covariance matrices to prevent singularities in cases where the data is sparse or high-dimensional.

In summary, by optimizing key parameters for each clustering method—such as the number of clusters, distance metrics, and regularization techniques—we aim to enhance the clustering results and validate the classification output from XLNet.

# Evaluation metric

To compare the performance of XLNet model, we will use standard evaluation metrics, including **accuracy**, **precision**, **recall**, and **F1 score** [4]. The comparison will allow us to assess the strengths and weaknesses of each model in distinguishing between mathematics and computer science texts, with a specific focus on how well they perform after the removal of domain-specific terminology. Additionally, we will use three key clustering evaluation metrics: **Silhouette Score**, **Calinski-Harabasz Index**, and **Davies-Bouldin Index**.

## Text Classification Metrics

**Accuracy**:  
Accuracy measures the proportion of correctly predicted instances (both true positives and true negatives) out of the total number of instances. It gives an overall view of how well the model is performing. However, it can be misleading if the dataset is imbalanced, as it doesn’t account for the distribution of classes.

**Precision**:  
Precision focuses on the positive predictions and measures how many of them are actually correct. High precision means that when the model predicts a positive class, it is usually correct. It’s useful when the cost of false positives is high.

**Recall** (also known as Sensitivity or True Positive Rate):  
Recall measures the ability of the model to correctly identify all relevant positive cases. It’s useful when missing a positive instance (false negative) is more costly than incorrectly labeling a negative instance as positive.

**F1 Score**:  
The F1 score is the harmonic mean of precision and recall. It provides a balanced view of the two metrics, especially in cases where there is an uneven class distribution. It is useful when we need a balance between precision and recall.

## Text Clustering Metrics

**Silhouette Score:**

The Silhouette Score measures how similar a data point is to the other points in its own cluster compared to points in other clusters. This score ranges from -1 to +1, with a score closer to +1 indicating that the clusters are well-separated and internally cohesive. A score of 0 suggests that the clusters are overlapping, and negative values indicate that points may be assigned to the wrong clusters. For each data point, the average distance to other data points within the same cluster is calculated (). This value represents the similarity level of the data point to others in its cluster. On the other hand, () represents the average distance to all other clusters it doesn’t belong to is computed. This value indicates how different the data point is from data points in other clusters.

**Calinski-Harabasz Index:**

Also known as the Variance Ratio Criterion, the Calinski-Harabasz Index evaluates clusters based on the ratio of between-cluster dispersion to within-cluster dispersion. Essentially, this metric measures how compact the clusters are and how well-separated they are from each other. A higher Calinski-Harabasz Index indicates that the clusters are dense and far apart, which is ideal for our project.

* **BGSS**: Between-group sum of squares (inter-cluster dispersion);
* **WGSS**: Within-group sum of squares (intra-cluster dispersion);
* **N**: Total number of observations.
* **K**: Total number of clusters.

**Davies-Bouldin Index:**

The Davies-Bouldin Index assesses the quality of clustering by calculating the average similarity ratio between each cluster and its most similar cluster. Unlike the other metrics, a lower Davies-Bouldin Index is better, as it indicates that clusters are compact and distinct from one another. The score takes into account both the cluster separation and the internal coherence of the clusters.

* is the number of clusters
* is the average within-cluster distance of cluster
* is the average within-cluster distance of cluster
* is the between-cluster distance of cluster and

# Challenges

Several challenges are anticipated throughout the course of this project, particularly given the novel approach of removing domain-specific terminology and relying solely on general linguistic patterns (metalanguage) for classification and clustering. These challenges span from dataset preparation to model optimization and clustering validation.

## Dataset Challenges

One of the key challenges is preparing the dataset, especially when working with academic PDFs. Some documents are scanned, meaning they consist of images rather than embedded text. The use of Optical Character Recognition (OCR) through Tesseract may introduce noise and errors, particularly when dealing with mathematical symbols, formulas, and non-standard characters. Additionally, filtering out irrelevant content (such as tables of contents, indices, or reference lists) without manual intervention is difficult. Ensuring that only relevant text remains for analysis will require a carefully fine-tuned algorithm, and errors at this stage could skew the results.

## Preprocessing Difficulties

Preprocessing is another complex phase of the project, involving multiple challenges:

* Irrelevant Pages:

Identifying and removing irrelevant sections, such as bibliographies and indices, is complicated by the inconsistent structure of academic books. Developing an automated process for this requires careful design to ensure useful content is not discarded.

* Character Cleaning:

Text extraction from scanned PDFs often introduces unwanted characters. Cleaning the text by filtering these irrelevant characters is crucial for ensuring the dataset is suitable for classification and clustering.

* Removing Domain-Specific Terms:

Building an effective dictionary of domain-specific words from both mathematics and computer science is critical to this project. This dictionary must capture a wide range of terms, including multi-word phrases, while ensuring the removal of domain-specific language does not interfere with the linguistic patterns necessary for accurate classification.

## Model Training and Hyperparameter Optimization

Training and fine-tuning XLNet for text classification presents its own set of challenges. Hyperparameters such as learning rate, batch size, and sequence length must be optimized for the dataset, and finding the best settings can be computationally intensive. Additionally, XLNet’s sensitivity to preprocessing, especially after the removal of domain-specific terminology, requires careful experimentation to avoid overfitting or underfitting. This is particularly important as XLNet relies on subtle linguistic cues in the absence of specialized vocabulary. Extensive experimentation with hyperparameters and training techniques will be necessary to achieve high accuracy.

## Classification and Clustering without Domain-Specific Terms

The core challenge of this project is accurately classifying texts after removing domain-specific terms. Traditional classification methods rely on key terms to distinguish between fields, but the removal of these terms forces XLNet to rely on subtler cues such as sentence structure, syntax, and function words. Achieving high classification accuracy in this context is difficult, and the risk of misclassifying closely related fields (e.g., Calculus and Probability) is significant. Testing different configurations, including stop-word removal and segmentation sizes, will be necessary to optimize performance.

Similarly, validating the classification through clustering methods like K-means, PAM, Density Clustering, and Gaussian Mixture presents its own challenges. The clustering results must align with the classification outputs to provide meaningful insights into how closely related fields are grouped, and tuning the parameters for each clustering technique will be crucial for achieving robust results.

# Testing Plan

To ensure our project reaches its highest potential, we have put together a straightforward testing plan. This plan outlines how we'll check the text preprocessing part of our project, from the initial stages of converting PDF’s data to the more complex phases of removing stop words. By following this plan, we can catch any issues early, make sure the data used in the project is reliable. The plan includes a table that details what we'll test, the expected outcomes, and how these tests fit into our overall goals.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test ID** | **Description** | **Objective** | **Method** | **Expected Outcome** |
| 1 | Text Extraction Accuracy | Evaluate the accuracy of text extraction tools on different PDF types | Extract text from a random sample of PDFs, compare to original PDF | High accuracy in text extraction |
| 2 | Domain Terminology Removal | Test the algorithm's ability to remove domain-specific terms | Use script to check if all stop words was removed | Text cleaned of any domain-specific terminology |
| 3 | Multi-word Term Removal Efficacy | Test the effectiveness of removing multi-word terms | Apply the dictionary-based removal method to texts with known terms | Accurate removal of multi-word terms |
| 4 | Efficiency of OCR on Scanned Texts | Measure the OCR accuracy on scanned texts | Apply OCR to a set of scanned pages, manually verify OCR output | High accuracy and minimal character misread |
| 5 | Data Source Validity | Ensure that all textual data comes from reputable and authoritative sources. | Ensure that all textual data comes from reputable and authoritative sources. | All sources are recognized academic or educational institutions. |
| 6 | Completeness of Domain-Specific Terminology Dictionary | Confirm the dictionary covers all relevant terms used within the academic texts. | Manually review a subset of texts to ensure all domain-specific terms are included in the dictionary. | No unlisted domain-specific terms are found in the reviewed texts. |
| 7 | Stop Word Removal Impact | Examine how the presence or absence of stop words affects the classification accuracy. | Compare model performance with and without stop word removal on a subset of the data. | Texts with stop words are easier to classify. |

# Bibliography

|  |  |
| --- | --- |
| [1] | Z. Yang et al., "XLNet: Generalized autoregressive pretraining for language understanding," in *Proc. 33rd Conf. Neural Information Process. Syst. (NeurIPS)*, 2019. |
| [2] | E. Schubert and P. J. Rousseeuw, "Faster k-medoids clustering: Improving the PAM, CLARA, and CLARANS algorithms," *Journal of Statistical Software,* 2022. |
| [3] | J.-O. Kapp-Joswig and B. G. Keller, "Clustering—Basic concepts and methods," *Journal of Chemical Theory and Computation,* 2021. |
| [4] | S. Minaee et al., "Deep learning based text classification: A comprehensive review," *ACM Computing Surveys (CSUR),* 2021. |
| [5] | T. Horáková, M. Houška, and L. Dömeová, "Classification of the educational texts styles with the methods of artificial intelligence," *Journal of Baltic Science Education,* 2017. |
| [6] | B. Touis, S. Warraki, O. El, and J. Lahiassi, "Using semantic networks for text classification in education: Generating tailored questions for students," *Proc. Int. Conf. E-Learning Smart Eng. Syst. (ELSES),* 2023. |
| [7] | Md. I. Hossain, H. U. Khan, M. S. Irfan, and M. R. Basir, "A survey on text classification algorithms from text to predictions," *International Journal of Innovative Research in Computer and Communication Engineering,* 2019. |
| [8] | S. Roy, "A survey on text classification from traditional to deep learning," *Journal of Computing Technologies,* 2020. |
| [9] | S. Vijayarani and S. Janani, "A survey on text classification algorithms," *nternational Journal of Computer Science and Information Technologies,* 2016. |
| [10] | Research Data Pod, "Paper Reading: XLNet Explained," 8 9 2023. [Online]. Available: https://researchdatapod.com/paper-reading-xlnet-explained/. [Accessed 21 9 2024]. |