MisMine: A Data-Driven Approach to Understanding Student Learning

CSCE-689 Programming LLMs Final Project Report

Yu-Sheng Chen (334003776)

Deliverables

- Github Repo: https://github.com/YSChen0609/Eedi-Competition/tree/main
- Demo Video: https://drive.google.com/file/d/1VbmRqk8_sCgZiPsD3CtTbkp0WSBtx8rl/view?usp=sharing
- Model Weights: https://drive.google.com/drive/folders/1s2CHHofe6MCfkIjog5i8tvY4fZtxksNY?usp=sharing
- main.ipynb
- Report.pdf

1. Introduction

1.1 Project Overview

This project addresses the EEDI “Mining Misconceptions in Mathematics” Kaggle competition, which focuses on identifying and analyzing mathematical misconceptions among students. The competition aims to improve educational outcomes by better understanding how students learn and where they commonly make mistakes.

1.2 Competition Background

The competition is hosted on Kaggle by EEDI, an educational technology company specializing in diagnostic questions and adaptive learning. The challenge involves analyzing student response data to identify patterns in mathematical misconceptions, with the ultimate goal of helping educators provide more targeted and effective instruction.

1.3 Problem Statement

The primary objective is to predict student responses to mathematical questions based on historical data, while identifying underlying patterns that might indicate common misconceptions. This involves: - Analyzing student response patterns - Identifying relationships between questions and mathematical concepts - Predicting student performance on future questions

The output of the project is to predict the MisconceptionId for each answer text.

2. Methodology

2.1 Data Description

The project utilizes dataset provided by EEDI, containing: - Student response data - Question metadata - Construct (concept) information - Student demographic information

The primary training data is stored in ‘train.csv’, which includes: - QuestionId: int (unique identifier for each question) - ConstructId: int (unique identifier for each construct) - ConstructName: str (name of the construct) - SubjectId: int (unique identifier for each subject) - SubjectName: str (name of the subject) - CorrectAnswer: str (correct answer to the question) - QuestionText: str (text of the question) - AnswerAText: str - AnswerBText: str - AnswerCText: str - AnswerDText: str - MisconceptionAId: int - MisconceptionBId: int - MisconceptionCId: int - MisconceptionDId: int

The MisconceptionDId is stored in the ‘misconception_mapping.csv’ file, which contains: - MisconceptionId: int (unique identifier for each misconception) - MisconceptionName: str (name of the misconception)

2.2 Data Preprocessing

- Drop QuestionId, ConstructId, SubjectId since they are not needed for the model.
- Transform the data into rows of ‘AnswerTexts’ and MisconceptionId so that we can use the embeddings of the ‘AnswerTexts’ to predict the MisconceptionId.
- Join the ‘misconception_mapping.csv’ on ‘MisconceptionId’ to get the ‘MisconceptionName’.
- Use pretrained embeddings to get the embeddings of the texts.
- The input tensor of the model will include:
    - ConstructName_embedding
    - SubjectName_embedding
    - QuestionText_embedding
    - CorrectAnswerText_embedding
    - AnswerText_embedding
    - MisconceptionName_embedding

2.3 Model Architecture

Since the competition introduction menttioned that pre-trained models are not performant enough (discuss later), I tried two kinds of Attention-based models: - Customized Attention-based model - Mixture of Experts (MoE) The details of the models can be found in the main notebook.

2.3.1 Customized Attention-based model

- Input Layer: Accepts embedded text data (input tensor) of dimension `input_dim`
- Attention Layer: Custom attention mechanism with 8 attention heads
- Dense Layers: Three-layer feed-forward network
    - First layer: 2048 units
    - Second layer: 1024 units
    - Output layer: Matches embedding size (768 dimensions)

2.3.2 Mixture of Experts (MoE)

- Number of experts: 8
- Top-k selection: **k=2**
- Components:
    - Individual expert networks
    - Gating network for expert selection
    - Layer normalization for stability
    - ReLU activation functions

Gating Mechanism - Input → hidden layer → expert selection - Top-k expert selection using softmax - Weighted combination of expert outputs

Forward Pass Process - Flattens input if needed - Computes gate logits - Selects top-k experts - Combines expert outputs weighted by gates

3. Experimental Design

1. Primary Metric
    - Top-5 accuracy for misconception prediction
    - Evaluates model's ability to identify correct misconception among top 5 predictions
2. Training and Validation Strategy
    - Training for 10000 epochs
    - MSE for loss function
    - 80% of the data for training, 20% for validation (test set is hidden by the competition)
3. Technical Environment
    - Kaggle Notebooks environment
    - NVIDIA T4 GPU support
4. Software Stack
    - Python programming language
    - Key libraries:
        - PyTorch for deep learning models
        - scikit-learn for evaluation metrics
        - pandas and numpy for data manipulation

4. Results and Discussion

4.1 Model Performance

Model	Top-5 Accuracy (Validation)
Customized Attention	16.70%
Mixture of Experts	14.42%

4.2 Performance Analysis

1. **Overall Performance**
    - Both models show relatively low accuracy, suggesting the challenging nature of misconception identification
    - The Customized Attention model outperforms the MoE model by approximately 2.28 percentage points

2. **Possible Explanations for Performance Difference**

    1. **Attention Mechanism Advantages**
    - Better capture of contextual relationships between question elements
    - More effective at handling the sequential nature of mathematical concepts
    - Uniform processing of all inputs, ensuring no information loss
    1. **MoE Limitations**
    - Top-k selection (k=2) might be too restrictive for this complex task
    - Expert specialization might not align well with the diverse nature of mathematical misconceptions
    - Potential information loss during expert selection process

3. **Contributing Factors to Low Overall Accuracy**
    - High complexity of mathematical misconceptions
    - Possible data sparsity for certain misconception types
    - Challenge of capturing subtle differences between related misconceptions
    - Limited context available in the text-only representation

4.3 Discussion and Potential Improvements

1. Is pre-trained models really not performant? The competition introduction mentioned that pre-trained models are not performant enough. However, I see many participants using pre-trained models to get a good score. This has inspired me that the competition introduction might not be completely true, since they might have not

used pre-trained models with other LLM techniques like prompt engineering or RAG.

2. I might have selected the wrong path to approach this problem. I simply tried to train a attention-based model to predict the MisconceptionId. However, the problem might be more complex than that. Using a pre-trained model with RAG might be a better approach to transfer the knowledge of the powerful pre-trained models.

3. How to improve the performance of my current models?

- Increase number of attention heads (Multi-head Attention)
- Add more layers to capture deeper relationships
- Implement cross-attention mechanisms
- Increase number of experts or top-k selection
- Implement hierarchical expert structure
- Add attention mechanisms within expert networks

5. Conclusions and Future Work

5.1 Summary

This project explored mathematical misconception identification using two approaches: a Customized Attention model (16.70% top-5 accuracy) and a Mixture of Experts model (14.42%). While demonstrating the feasibility of automated misconception detection, results indicate significant room for improvement.

5.2 Key Findings

- Attention mechanisms show promise in capturing mathematical relationships
- Complex architectures like MoE require more sophisticated tuning
- Text embeddings can effectively represent mathematical content
- Current accuracy levels suggest need for enhanced approaches

5.3 Future Work

- Experiment with pre-trained models with RAG (or other LLM techniques)
- Implement hybrid attention-expert models
- Explore ensemble methods
- Enhanced feature engineering

References

1. EEDI Competition Page: https://www.kaggle.com/competitions/eedi-mining-misconceptions-in-mathematics
2. My work is highly inspired by the kaggle notebook: https://www.kaggle.com/code/nishantjoshi7/eedi-project